TECHNISCHE
UNIVERSITÄT
DRESDEN

Faculty of Business and Economics Chair of Business Management, esp. Industrial Management

Energy aware hybrid flow shop scheduling

Sven Schulz

Born on: 28th July 1990 in Salzwedel

Dissertation

to achieve the academic degree Doctor rerum politicarum (Dr. rer. pol.)

First referee

Prof. Dr. Udo Buscher

Second referee

Prof. Dr. Rainer Lasch

Submitted on: 30th April 2020

Table of Contents

List of Figures

List of Tables

List of Algorithms

List of Abbreviations

DMS	Deterministic Machine Selection
DPSO	Discrete Particle Swarm Optimization
EAS	Energy Aware Scheduling
ECS	External Conversion System
ECT	Earliest Completion Time
EES	Energy Efficient Scheduling
FFSP	Flexible Flow Shop Problem
FLX	Flexible Processing Energy Demand
GA	Genetic Algorithm
HFS	Hybrid Flow Shop
HFSSP	Hybrid Flow Shop Scheduling Problem
HPSO	Hybrid Particle Swarm Optimization
HV	Hypervolumn
ILS	Iterated Local Search
LMS	Leveling Machine Selection
LS	Local Search
MIP	Mixed-Integer Program
MILP	Mixed-Integer Linear Program
MMRCPSP	Multi-Mode Resource-Constrained Project Scheduling Problem
NAM	Not All Machine problems
NBC	Number of Continuous Variables
NBV	Number of Binary Variables
NC	Number of Constraints
NCN	Number of Considered Neighbors
NDS	Non-Dominated Solutions
NSGA-II	Non-Dominated Sorting Genetic Algorithm-II
POX	Position Based Crossover Operator
PS	production System
PSO	Particle Swarm Optimization

RPS	Reference Point Sharpness
RTP	Real-Time Prices
SD	Standard Deviation
TEC	Total Energy Costs
TOU	Time-Of-Use (energy prices/ tariffs)
TS	Tabu Search
TT	Total Tardiness
TWT	Total Weighted Tardiness

List of Symbols

Chapter 2

C_{max}	Maximum completion time (makespan)
F_m	Flow shop with m machines
HF	Hybrid flow shop
J_m	Job shop with m machines
$P2$	Parallel machine problem with two machines
p_j	Processing time of job j
S	Solution space of an optimization problem
T_j	Tardiness of job j
x	Single solution of an optimization problem
Z^i	Objective function values of pareto optimal solution i
Z^{i*}	Lexicographic solution i
α	Machine environment
β	Job characteristics
γ	Objectives
$\zeta_i(x)$	Objective i depending on solution x

Chapter 3

$c_{jk} \in \mathbb{N}^+$	Completion time of task k of job j
D_j	Due date of job j
$e^t_{jk} \in \mathbb{R}^+$	Energy consumption of task k of job j at time-interval t
ec^t	Electricity cost during time period t
$EC_{jk} \in \mathbb{R}^+$	Energy costs of task k of job j
ed_{jk}	Maximum energy demand at maximum speed of task k of job j
$EP_{jk} \in \mathbb{R}^+$	Energy consumption of task k of job j
es_{jkl}	Energy consumption of task k of job j at the speed reduction l

es^*_{jkl}	Relative energy saving if speed is reduced from $(l-1)$ to l
$g_{jkl} \in \{0,1\}$	The speed reduction l of task k of job j is set as individual variable
$g^*_{jkl} \in \{0,1\}$	The speed reduction l of task k of job j is set as special ordered set
i	Index for machine
j	Index for jobs
$\mathcal{J}$	Set of n jobs
k	Index for production stage or task
l	Level of speed reduction as additional processing time
m	Number of stages
$\mathcal{M}_k$	Set of parallel machines at stage k
n	Number of jobs
o	Number of speed reduction levels
p_{jk}	Minimum processing time of task k of job j
P_{jk}	Actual processing time of task k of job j
$\mathcal{S}$	Set of m stages
$s_{jk} \in \mathbb{N}^+$	Start time of task k of job j
t	Index for discrete time-intervals
T	Set of discrete time-intervals
$T_j \in \mathbb{N}^+$	Tardiness of job j
TEC	Total Electricity Costs
tec^t_{jkl}	Energy costs for job j at stage k, if processing starts in time period t at level of speed reduction l
TT	Total Tardiness
$\mathcal{V}$	Set of speed reduction levels
$x^t_{jk} \in \{0,1\}$	Task k of job j is performed at time t
$x_{jj'k} \in \{0,1\}$	Job j starts after j' at stage k.
$y_{jki} \in \{0,1\}$	Job j is executed by machine i at stage k
$z^t_{jk} \in \{0,1\}$	Execution of task k of job j starts at time t
$z^t_{jkl} \in \{0,1\}$	Execution of task k of job j starts at time t with speed l
μ_k	Number of parallel machines at stage k
τ	Number of time-intervals

Chapter 4

A	The non-dominated solution set found
$a^t_{jk} \in \{0,1\}$	Task k of job j is processed in time t

$b_{jk}^{t} \in \{0,1\}$	Task k of job j starts in time t
$C_{jk} \in \mathbb{N}^{+}$	Completion time of task k of job j
c_1	First acceleration coefficient
c_2	Second acceleration coefficient
$C(A,B)$	Coverage metric between solution sets A and B
CT	Converging time
d_j	Due date of job j
$d(a,r)$	Normalized euclidean distance between a and r
e_{jk}	baseline energy consumption of job j at stage k
$ec_{jk}^{t} \in \mathbb{R}^{+}$	Energy consumption of task k of job j at time-interval t
ep^{t}	Electricity price per kWh in time period t
es_{jk}	Energy saving factor depending on v_{jk} ($es_{jk} \in [0,1)$)
F_1	Mutation Operator
F_2	Crossover Operator
G^{t-1}	Global optimum of all particles during the past $t-1$ iterations
$GD(A,B)$	Generational distance between solution set A and reference front R
$gbest$	Best previous global optimum of all particles
i	Indes for particle
I_{max}	Maximum number of iterations in the tabu search
j	Index for jobs
k	Index for production stage or task
l	Index for parallel machines
m	Number of stages
m_k	Number of parallel machines at stage k
n	Number of jobs
$NNDS$	Number of non-dominated solutions
o_{jk}	Operation of job j at stage k
p	Number of particles/ swarm size
p_{jk}	Baseline/ minimum processing time of job j at stage k
P_{jk}	Actual processing time of job j at stage k
P_i^{t-1}	Local optimum of particle X_i during the past $t-1$ iterations
$pbest$	Best previous local optimum of a particle
R	Reference front (best known pareto front)
$S[j][l][k]$	Sequencing and machine assignment information of a solution
t	Index for discrete time-intervals or iterations in the heuristic
$T_j \in \mathbb{N}^{+}$	Tardiness of job j

TEC	Total Electricity Costs
TT	Total Tardiness
$\mathcal{V}$	Set of v_{max} possible time increases
v_{max}	Maximum additional processing time
v_{jk}	Processing time increase of job j at stage k
$V[j][l][k]$	Additional processing time information of a solution
X	Tupel representing a solution
X_i^t	Particle i in iteration t
δ_i^t	Intermediate particle i after crossover
θ	Tabu tenure
λ_i^t	Intermediate particle i after mutation
τ	Number of time-intervals in the observation period
ω	Inertia weight for the PSO

Chapter 5

a_s^{mv}	Energy savings on machine m at stage s for speed v
$c_{sj} \in \mathbb{N}$	Completion time of task s of job j
ce_t	Electricity cost in time period t
cp_j	Production cost of job j
ct_j	Tardiness cost of job j
D_j	Due date of job j
E_{sj}^m	Energy consumption of job j on machine m at stage s
$g_{sj}^{mv} \in \{0,1\}$	Processing time extension v of task s of job j on machine m
j	Index for jobs
J	Set of n jobs
m	Index for machines
M_s	Set of Machines at stage s
n	Number of jobs
P_{max}	Maximum peak power
$p_{sjt}^m \in \mathbb{N}$	Power consumption of task s of job j on machine m in time period t
R_j	Release date of job j
s	Index for stages
S	Set of stages
S_{sj}^m	Standard processing time of job j on machine m at stage s
t	Index for time periods

T	Set of time periods
$T_j \in \mathbb{N}$	Tardiness of job j
v	Index for speed level
V	Set of speed levels
$x_{sjt}^m \in \{0, 1\}$	Task s of job j is performed on machine m in time period t
$y_{sj}^m \in \{0, 1\}$	Task s of job j is assigned to machine m
$z_{sjt}^m \in \{0, 1\}$	Execution of task s of job j on machine m starts in time period t

Chapter 6

a_{sj}^m	Total energy demand for processing job j on machine m at stage s
C_{max}	Maximum completion time (Makespan)
$c_{sj} \in \mathbb{N}$	Completion time of task s of job j
E_{sj}^m	Power consumption of job j on machine m at stage s per time period
j	Index for jobs
J	Set of n jobs
J_{max}	Number of jobs
m	Index for machines
M_s	Set of heterogeneous Machines at stage s
$M_{s,max}$	Number of heterogeneous Machines at stage s
n	Incumbent solution of the local search
n^*	Neighbor solution with best fitness
$n^{'}$	Perturbed solution
NCN	Number of considered neighbors
PP	Peak Power
$p_{sjt}^m \in \mathbb{N}$	Power consumption of task s of job j on machine m in time period t
P_j	Total processing time of job j
P_{sj}^m	Processing time of job j on machine m at stage s
RPS	Reference point sharpness
RTP_t	Real time price for energy consumption in time period t
s	Index for stages
S	Set of stages
S_{max}	Number of stages
$SMAP_{sj}^m$	Normalized total energy demand
t	Index for time periods
T	Set of time periods

TEC	Total energy costs
X	Random number
$x^m_{sjt} \in \{0,1\}$	Task s of job j is performed on machine m in time period t
$y^m_{sj} \in \{0,1\}$	Task s of job j is assigned to machine m
$z^m_{sjt} \in \{0,1\}$	Execution of task s of job j on machine m starts in time period t
α	Stretch factor to define upper bounds for completion times
η^m_{sj}	Probability that job j is allocated to machine m at stage s
μ	Number of assigned machine
Π	Permutation of jobs

Chapter 7

a_{il}	Machine-hour rate for machine l at stage i
b_i	Number of external machines at stage i
c_i	Transportation charge at stage i
$C_{ij} \in \mathbb{N}$	Completion time of stage i of job j
e_t	Energy cost at time t (real time price)
e	Index for external machines
E_i	Set of external machines at stage i
i	Index for stages
I	Set of stages
j	Index for jobs
J	Set of jobs
k	Index for in-house machines
K_i	Set of in-house machines at stage i
l	Index for all machines
L_i	Set of all (o_i+b_i) machines at stage i
n	Number of jobs
o_i	Number of in-house machines at stage i at stage i
p_{ijl}	Processing time of job j at stage i on machine l
s	Number of stages
t	Index for time periods
T	Set of time periods
T_{max}	Number of time periods
v_ik	Energy demand of machine k at stage i
$X_{ijlt} \in \{0,1\}$	Job j is produced on machine l at stage i in time period t

$Y_{ijl} \in \{0,1\}$	Job j is manufactured on machine l at stage i
z_i	Transportation time at stage i in case of subcontracting
$Z_{ijlt} \in \{0,1\}$	Job j is started on machine l at stage i in time period t

Chapter 8

a_{il}	Machine-hour rate for machine l at stage i
b_i	Number of external machines at stage i
c_i	Transportation charge at stage i
$C_{ij} \in \mathbb{N}$	Completion time of stage i of job j
e_t	Energy cost at time t (real time price)
e	Index for external machines
E_i	Set of external machines at stage i
i	Index for stages
I	Set of stages
j	Index for jobs
J	Set of jobs
k	Index for in-house machines
K_i	Set of in-house machines at stage i
l	Index for all machines
L_i	Set of all (o_i+b_i) machines at stage i
m	Number of stages
n	Number of jobs
o_i	Number of in-house machines at stage i at stage i
p_{ijl}	Processing time of job j at stage i on machine l
t	Index for time periods
T	Set of time periods
v_ik	Energy demand of machine k at stage i
$X_{ijlt} \in \{0,1\}$	Job j is produced on machine l at stage i in time period t
$Y_{ijl} \in \{0,1\}$	Job j is manufactured on machine l at stage i
z_ie	Transportation time at stage i in case of subcontracting to e
α	Crossover points depending on production stages
γ	Maximum number of generations without improvement
η_m	Mutation probability
τ	Number of time periods

1 Introduction

1.1 Motivation

"Only if humanity acts quickly and resolutely can we limit global warming"[1] conclude more than 25,000 academics with the statement of SCIENTISTS FOR FUTURE. The concern about global warming and the extinction of species has steadily increased in recent years. It has become more and more present in everyday life due to protests like the FRIDAYS FOR FUTURE movement. It is important to question the greenhouse gas emissions and with them the global energy consumption. The industrial sector is responsible for around half of the world's primary energy requirements,[2] which is why manufacturing companies in particular are under increasing political and social pressure to intensify their sustainability efforts. How industrial companies can influence their energy consumption through targeted operational planning will be the core question of this doctoral thesis.

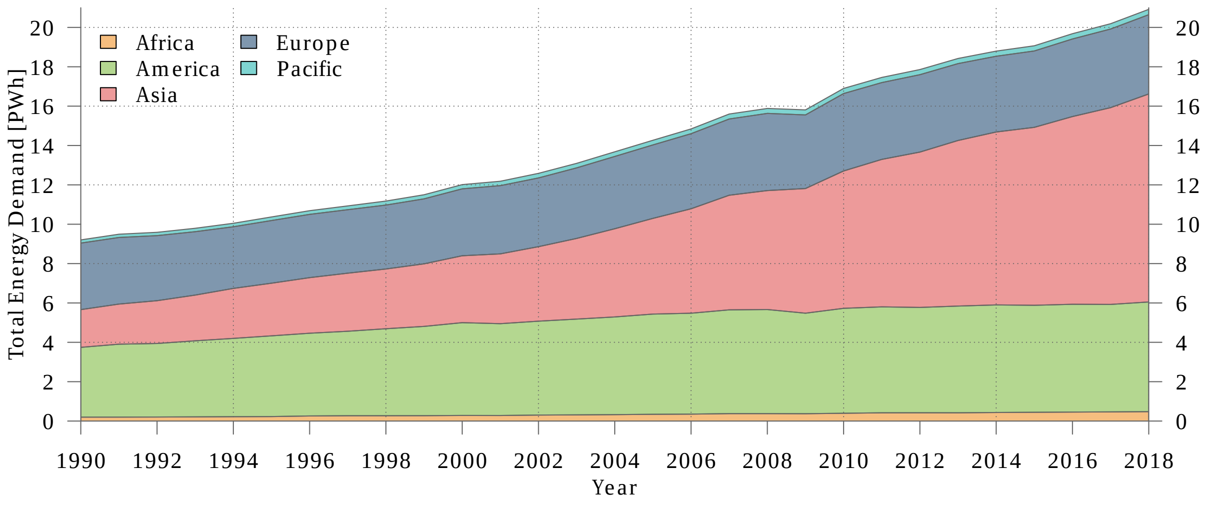


Figure 1.1: Development of electricity consumption since 1990 in PWh[3]

A large part of industrial energy demand is covered by electricity. The demand is constantly increasing due to automation and digitalization. The increase in electricity

[1]HAGEDORN et al. (2019): *Concerns of young protesters are justified*, p. 140.
[2]Cf. BP ENERGY ECONOMICS (2019): *Energy Outlook 2019*, p. 15.
[3]Own illustration. Data from ENERDATA (2019): *Global Energy Statistical Yearbook 2019*.

consumption over the last 30 years is shown in Figure 1.1. Apart from the 2008/2009 financial crisis, consumption has increased every year. The main reason for the sharp rise is the increased demand in emerging and developing countries. As a result, strong rises have been recorded above all in Asia. In Europe, on the other hand, the volume has remained rather constant.

Figure 1.2 shows the worldwide energy requirements in more detail and country by country. The size of each bubble represents the energy consumption of a country in 2018 and the x-axis shows the change in energy consumption compared to 2017. While China as the worldwide biggest energy consumer shows a growth rate of 7.7 %, European countries like France and Germany were able to reduce total consumption by around 1 % despite economic growth.

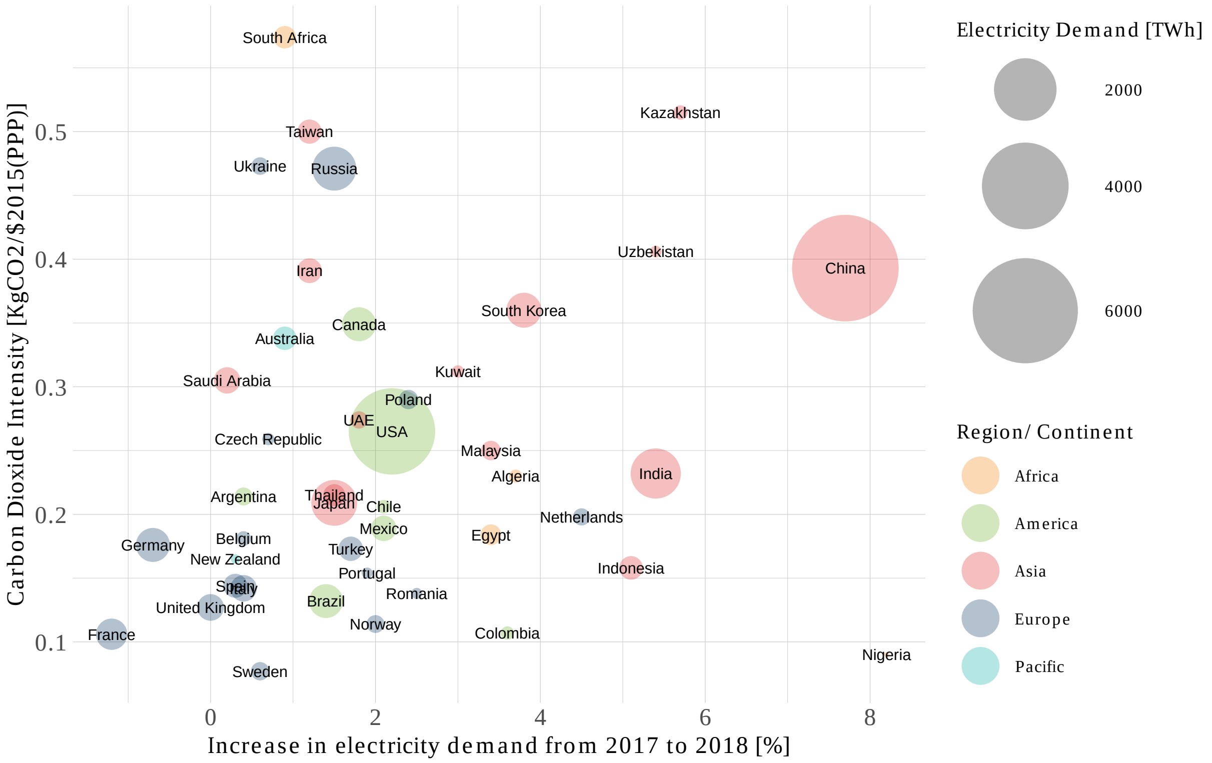


Figure 1.2: Summary of Energy Consumption in 2018[4]

It is well known that countries differ widely in their primary energy consumption, greenhouse gas emissions and energy efficiency. The y-axis in Figure 1.2 shows how many kg of CO_2 certain countries emit in order to generate 1 $ (2015PPP) of their GDP.[5] While Sweden emits less than 100 g CO_2 to produce goods worth $1, South Africa causes more

[4]Own illustration. Data from Enerdata (2019): *Global Energy Statistical Yearbook 2019*, Note that countries with very low consumption or without reliable data are not shown.

[5]More precisely, the ratio of gross domestic product to CO2 emissions is shown here. Since more recent data are not available, the reference is set to the purchasing power parity of the year 2015.

than 550 g. The reasons are manifold and depend, among other things, on the energy mix, technological status, meteorological conditions and the share of economic sectors. Overall, no correlation can be ascertained between the increase in consumption and CO2 efficiency.[6] But it can again be identified that industrialized countries and especially Central European countries keep consumption constant in recent years and are more CO2 efficient than developing countries in Asia or Africa. Graphically speaking, they are located further down on the left.

In light of climate change mitigation efforts, countries should improve their CO_2 efficiency, leading them to affectively move downward in figure 1.2. Just as the causes for energy efficiency are multivariate, various approaches must also be pursued for reduction of energy consumption. Jiang et al. (2018) consider the following three approaches to be particularly important on the way to a more sustainable economy:[7]

1. Large share of renewable energy in electricity generation,[8]
2. Highly energy efficient technologies,
3. Increasing degree of electrification.

All of these approaches require both society and industry to invest heavily in new technologies and equipment. In addition to this strategic level, a rethink must also take place on the operational level within companies. During production planning and control, energy restrictions must also be taken into account to a greater extent. An increase in the share of renewable energies leads to more fluctuations in the electricity supply. Besides energy storage systems, this fluctuation must be absorbed by flexible load management on demand side.[9] Production scheduling plays an important role for manufacturing companies when adjusting their energy consumption to the supply. At the same time, scheduling can make a contribution to energy efficiency by reducing the consumption directly.

This thesis is dedicated to the methods and possibilities of **energy-aware production scheduling (EAS)**. EAS can not only make a significant contribution to energy efficiency in industry. In contrast to expensive investments in new technologies, operational planning approaches **require little monetary input** and **can be integrated quickly**.[10] Even if it would be desirable, companies do not reduce their energy consumption solely on the basis of environmental considerations. At the end of the day, production processes must

[6] Correlation coefficient is just 11.97 %.
[7] Cf. Jiang et al. (2018): *Transition scenarios of power generation*, p. 482.
[8] The authors also suggest nuclear power and biomass processes with negative emissions.
[9] Cf. Anke et al. (2018): *Lastverschiebepotentiale in Dresden*, p. 3.
[10] Cf. Zhou / Liu (2019): *Energy-efficient multi-objective scheduling*, p. 1282.

also be profitable. Although energy costs are the main objective in the following, other capacity and time-oriented criteria must be taken into account to ensure an economic production. Thus, it is important to consider multi-criteria optimization problems, which is the object of this thesis.

1.2 Problem definition

The area of green scheduling has received enormous attention in the research landscape over the last 10 years.[11] Consequently, the focus of this thesis is set to a specific problem setting, namely **hybrid flow shop (HFS)** problems. In classical flow shops all jobs follow the same production sequence and exactly one machine is available for each processing step. In practice, however, processing tasks are often of varying lengths, which means that additional capacities are created at bottleneck stages. A flow shop with parallel machines at different production stages is called HFS and is very common in practice as for example in semiconductor, electronics, paper or textile industry.[12] In this thesis the following **three problem settings** will be examined, which have not yet been investigated so far, but do show practical relevance:

HFS1: In times of highly networked supply chains and just-in-time deliveries, punctual completion is becoming increasingly important. Interestingly, in HFS literature minimizing tardiness and energy costs are hardly considered simultaneously as objectives. For that reason, the relationship between these two criteria shall be analysed in more detail. Additionally, variable production speeds and time-of-use electricity prices are taken into account for the first time.

HFS2: Besides the consumption charge for the actual demand in kWh, companies often pay a so-called demand charge for the maximum peak power in kW during the billing period. Thus, there are two approaches to reduce electricity costs while demand remains constant. On the one hand, electricity consumption can be levelled (demand charge) and on the other hand, loads can be shifted to times with lower energy prices. Obviously, both approaches are contradictory and it is questionable how both strategies are compatible. HFS2 considers both, power peaks and time-dependent electricity prices, as objectives simultaneously. In addition, the capacitive utilization is optimized in order to plan efficiently not only from an energetic perspective. As far as known to the author, these three

[11] See e.g. Gahm et al. (2016): *Energy-efficient scheduling.*

[12] Cf. Ruiz / Vazquez-Rodriguez (2010): *The hybrid flow shop scheduling problem*; Low / Hsu / Su (2008): *A two-stage hybrid flowshop scheduling problem.*

criteria and their interdependencies have not yet been analysed in HFS literature.

HFS3: Thirdly, the influence of subcontracting possibilities on EAS decisions will be examined. Companies may have several locations in different countries that can manufacture the same products. External outsourcing of production steps is another possibility. So far, the EAS literature does not take into account that individual processing steps can be subcontracted. Outsourcing energy-intensive jobs to other production sites is particularly useful if these facilities have lower energy costs, own electricity generators or rely on a more economic/ecological energy mix.

These three EAS problems shall be intensively examined in this thesis. Thereby, the following research questions will be addressed and discussed:

Q1: What are the main approaches in energy aware scheduling and what is the current state of research?

Q2: To what extent can electricity costs be reduced by changing production speeds and deliberate delays and how can that problem be mathematically formalized?

Q3: Which heuristic is suitable to solve a hybrid flow shop scheduling problem with variable execution modes and total tardiness as well as energy costs as objectives?

Q4: How do capacitive scheduling criteria interact with peak power and energy costs as additional objective functions?

Q5: Is an iterated local search algorithm suitable to find pareto optimal solutions in a three-objective energy aware hybrid flow shop problem?

Q6: Which influence has subcontracting on energy aware scheduling?

1.3 Structure of this work

To answer these six research questions, this work is divided into nine Chapters. An overview of the structure gives Figure 1.3. After the introduction motivates and delimits the problem, the second Chapter essentially explains the basics, which are indispensable for the understanding of this work. This includes theoretical fundamentals of HFS scheduling, general conditions in energy markets and an insight into the methods of multi-criteria optimization.

The main part of the work in Chapters 3 to 8 is devoted to the research questions. Each of the three presented problems HFS1, HFS2 and HFS3 is analysed in two sections. First, the problems are defined and examined in each case theoretically by setting up a MIP formulation. By means of the models and commercial solvers, exact solutions for small

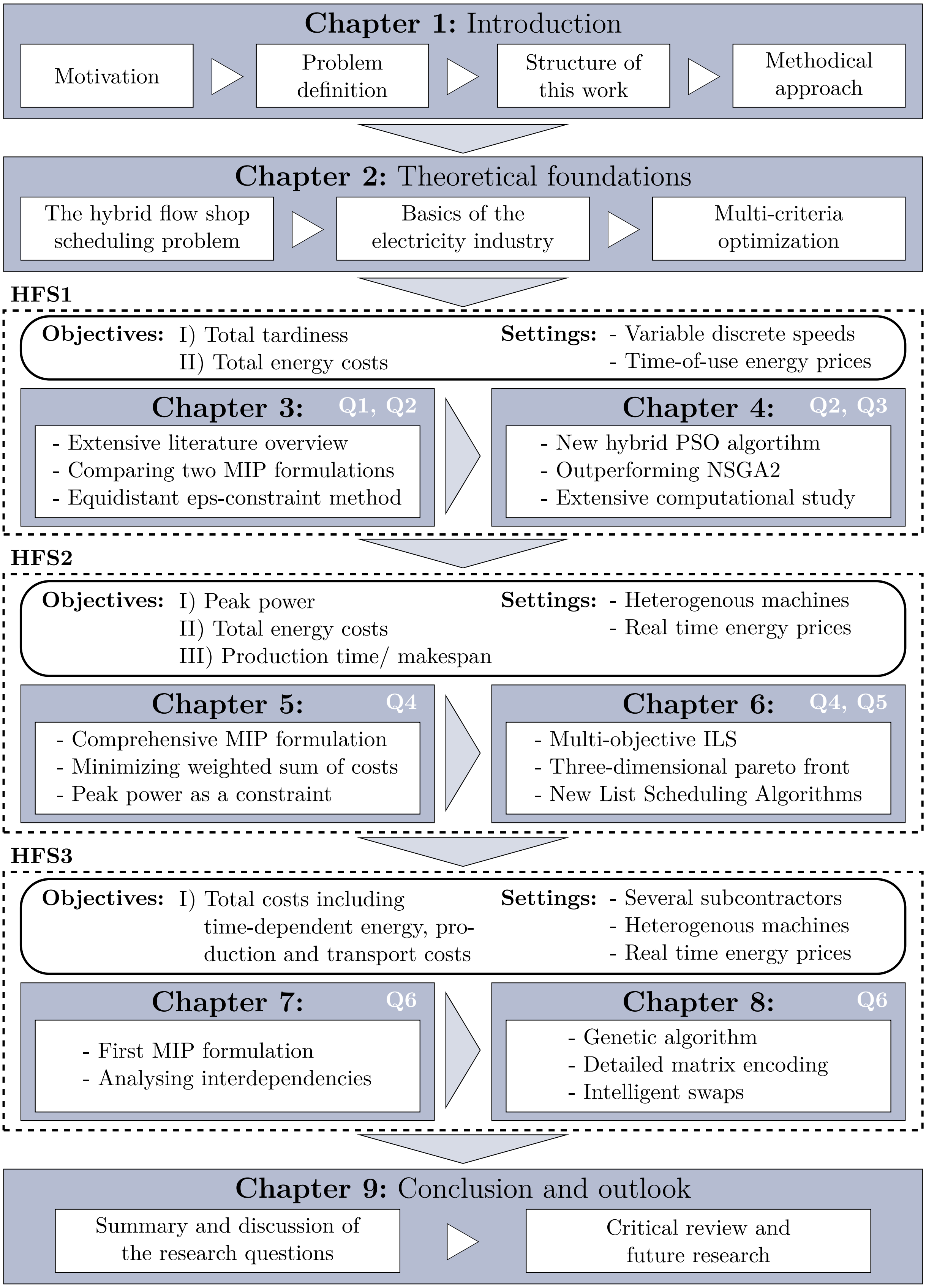


Figure 1.3: Structure of this work

problem instances can be calculated to analyse the properties of the problem. This is not only helpful for deriving recommendations for managerial activities in practice, but can also be useful to develop problem-specific heuristics. The development of such non-exact solution methods is done for each problem in a second Chapter. Such heuristic approaches allow to solve larger problems, which are of more practical relevance.

In detail, Chapter 3[13] and 4[14] investigate problem **HFS1**. First, an insight into the main approaches of EAS will be given and in this context the research question **Q1** will be discussed. Subsequently, Chapter 3 presents two different model formulations for an HFS with variable speeds for simultaneous minimization of total tardiness and total energy costs under time-dependent energy costs. The time-indexed as well as the sequence-indexed formulation are then examined with respect to problem size and computational complexity. Thereby, both MIP formulations show advantages and disadvantages. The equidistant epsilon-constraint method is used to optimize smaller test instances. All that is done to provide partial answers to research question **Q2**.

The first part of **Q2** is also discussed in Chapter 4 which will mainly resolve question **Q3**. To this end, a multi-objective hybrid particle swarm optimization approach is presented to solve problem **HFS1**. The new method not only outperforms classic benchmark algorithms like Non-dominated sorting genetic algorithm II (NSGA-II), but is also superior compared to already existing particle swarm based methods. Two main reasons for the good performance are the efficient de- and encoding procedure and the integration of a tabu search to intensify scanning of the solution space. It can be shown that substantial energy cost reduction can be achieved through intelligent scheduling without increasing the total tardiness.

The second problem **HFS2** is first approached in Chapter 5[15]. A new time-indexed MIP formulation is presented to minimize production and energy costs. The peak power consideration is added as a constraint and first answers to question **Q4** can be justified by using parametric optimization. However, the model is not yet a multi-criteria approach.

This will be changed in Chapter 6[16] where the model is rewritten with three separate objective functions. To solve the problem again an epsilon-constraint approach is used, which is integrated in IBM ILOG CPLEX. This allows small instances to be optimally solved and properties to be worked out. Since even simple forms of the HFS are NP-hard[17], it is inevitable to use a heuristic solution approach. For this purpose, an iterated local

[13] Chapter 3 is based on SCHULZ / BUSCHER / SHEN (2020): *Multi-objective hybrid flow shop scheduling.*
[14] Chapter 4 is based on DING et al. (2020): *Energy Aware Scheduling in Flexible Flow Shops.*
[15] Chapter 5 is based on SCHULZ (2018): *A Multi-criteria MILP Formulation.*
[16] Chapter 6 is based on SCHULZ / NEUFELD / BUSCHER (2019): *Comprehensive energy-aware hybrid flow shop.*
[17] Cf. GUPTA / HARIRI / POTTS (1997): *Scheduling a two-stage hybrid flow shop*, p. 173.

search is proposed according to **Q5**, which is implemented via a first stage permutation coding. Both machine allocation and sequencing on the further stages are conducted by constructive list scheduling approaches. Therefore, three different problem-specific algorithms are integrated to search the solution space in a diversified manner, leading to significantly better results than the NSGA-II.

The last research question **Q6** is discussed in Chapters 7[18] and 8[19]. Again, we start by setting up a model which allows to solve problem **HFS3** optimally. This time, a single cost function is considered as objective. Chapter 8 then presents a genetic algorithm to solve the problem, using a sophisticated matrix encoding. This is necessary to consider all theoretically possible solutions in the algorithm.

Finally, this work will be concluded in Chapter 9. In particular, the research questions will be discussed in detail once again and answers will be provided. In addition, the findings are critically reviewed and recommendations for future research in the field of energy-aware HFS are derived.

1.4 Methodical approach

In a driving car only the wheels have contact with the ground. No matter how well the vehicle was motorized, even if it was equipped with the latest technologies, it would still be difficult to drive and could even be dangerous if the tyres were not of high quality and correctly fitted. The same applies to research. If the method is not chosen correctly or the implementation is faulty, no valid findings can be derived. Analogous to the construction plan of a car, we need a research design[20] with coordinated methods to be able to answer the research questions as precisely as possible. Consequently, the methodical approach will be briefly discussed at this point.

First, however, this work is to be placed in the scientific context. Certain scholars argue that research can initially be divided into basic (theoretical) and applied (practical) science.[21] The aim of this thesis is to reproduce practical operational problems as accurately as possible and to develop solution approaches for them. Consequently, this work is classified as applied research. In particular, it is a problem of management sciences. Broken down even further, scheduling is part of operations management which includes all strategic, tactical and operational tasks that are necessary to produce goods or services with the

[18]Chapter 7 is based on Schulz / Apelmeier / Buscher (2017): *Hybrid Flow Shop Scheduling with Subcontracting Options.*

[19]Chapter 8 is based on Schulz (2019): *A genetic algorithm to solve the hybrid flow shop.*

[20]The term "reserach design" is used very differently in literature. Here it is based on Kirshenblatt-Gimblett (2006): *What is research design.*

[21]See e.g. Roll-Hansen (2009): *Distinction between basic and applied research.*

most efficient use of resources.[22] In summary, this work can thus be assigned to operations management, which is a part of management science that belongs to the applied sciences.

With regard to the research method, approaches can in principle be divided into quantitative and qualitative. In the field of operations management, quantitative approaches are not only used more frequently; they also form the origin of research in the field of operations and are labelled as operations research (OR), especially in Europe.[23] OR serves to support (especially economic) decision-making problems with the help of quantitative mathematical methods. Ideally, the optimal choice for a decision problem can be identified by means of OR. In addition to mathematics and management science, OR also integrates IT approaches. Due to the complexity of most decision problems, solutions can only be found with the help of efficient algorithms implemented in high-performance computer technology.

Within the OR there are various sub-disciplines. In this thesis, mainly approaches of mixed-integer optimization and heuristic solution approaches are used. These methods are combined with the knowledge of multi-criteria optimization, which will be described in detail in Section 2.3. The three identified problems **HFS1**, **HFS2** and **HFS3** will all be addressed in a similar way. The following procedure should enable valid results.

1. A formal description is made by a **mixed integer problem formulation**. This clearly defines the problem and distinguishes it from related problems.

2. Using **Branch and Cut Algorithm** of standard solvers, small instances can be optimally solved and properties worked out. For this purpose each model is implemented in IBM ILOG CPLEX.

3. A **(meta-)heuristic solution approach** is developed to solve larger and thus practice-relevant problems. Thereby, problem-specific properties are taken into account in order to enable an efficient search of the solution space.

4. Extensive **numerical study** can be used to check the suitability of the newly developed heuristic and to investigate the interdependencies of the different variables and objectives. Instructions for action can be derived from this. The test data is based on modified existing instances if available combined with real market data. Thus, the reality can be reflected as good as possible.

This methodical approach should enable a valid treatment of the topic. Besides the choice of methods and the classification into the research area, some further characteristics

[22]Cf. Lee (2018): *A review of applications of genetic algorithms in operations management*, p. 1.
[23]Cf. Bertrand / Fransoo (2002): *Operations management research methodologies*, p. 241.

of this thesis can be classified. Therefore, Table 1.1 summarizes the procedure according to the "research onion" suggested by Saunders / Lewis / Thornhill (2019).[24] The research onion is designed as a tool for the economic sciences to classify the methods step by step (symbolically layer by layer of an onion) and thus define the research design. In total, six different layers are considered, as listed in Table 1.1 from top to bottom (from the outside to the inside of the onion). In general, this approach is intended for empirical and qualitative research, since the authors mainly discuss data collection and analysis. Consequently, data is placed in the middle of the onion. This work is mainly dedicated to exact and heuristic OR methods and whether they are suitable for a specific problem. Nevertheless, the basic considerations can be transferred.

Layer	Classification	Explanation
Philosophy	Pragmatism	The optimization problem is in the centre and should be solved in the best possible way. Based on this, it can be applied in practice and act as a decision support.
Approach to theory development	Deduction	Large amount of data is used to evaluate the performance of the solution approaches. Furthermore, the interdependencies between decisions and objectives are analysed.
Methodical choice	Multi-method quantitative	Various quantitative OR methods like MIP formulations, branch & cut algorithm and heuristic approaches are used.
Strategies	Survey & Case Study	First, the literature is intensively analysed to filter out promising approaches. The development, parameter tuning and evaluation of the algorithms is done in case studies.
Time horizon	Longitudinal	Within a period under consideration (a day or week), planning should be as efficient as possible.
Data	Collection & Analysis	Where possible, market data is used. Otherwise, data is generated randomly according to the literature.

Table 1.1: Classification of research design using the research onion

[24]Cf. Saunders / Lewis / Thornhill (2019): *Research methods for business students*, pp.128.

2 Theoretical foundations

2.1 The hybrid flow shop scheduling problem

The underlying problem of scheduling can be defined as:

> *"...a decision-making process that is used on a regular basis in many manufacturing and services industries. It deals with the allocation of resources to tasks over given time periods and its goal is to optimize one or more objectives."*[1]

This dissertation deals with **production scheduling** and is therefore to be distinguished from other scheduling problems such as project scheduling, CPU scheduling or operating room scheduling in hospitals. However, solution methods are very similar and can certainly be transferred. Production scheduling is the last operative decision in production planning and control. It aims to determine the sequence of jobs in production and, based on this, the time allocation of jobs to the production resources (hereinafter referred to as machine).[2]

Production scheduling problems exist in various forms. To classify the different problems, Graham et al. (1979)[3] have introduced a notation with three fields $\alpha|\beta|\gamma$. This notation is also suitable for the problems considered here:

$$
\begin{aligned}
\text{Chapter 3 \& 4:} &\quad HFS|P_{jk}(v_{jk}), d_j, TOU|TT, TEC \\
\text{Chapter 5:} &\quad HFS|r_j, RTP|TC \\
\text{Chapter 6:} &\quad HFS|RTP|C_{max}, PP, TEC \\
\text{Chapter 7 \& 8:} &\quad HFS|t_{jk}, RTP|TC
\end{aligned}
$$

The first entry α describes the **machine environment**. The simplest problem consists of a single machine ($1||$). In this thesis **hybrid flow shop problems** ($HFS||$) are examined which is exemplified in Figure 2.1. All jobs pass the production stages in the same order which is initially referred to as a flow shop ($F||$). Additionally, more than one machine is available for at least one processing step. Scheduling a single task on multiple resources is called a parallel machine ($P||$) problem. Thus, as shown in Figure 2.1, HFS

[1] Pinedo (2012): *Scheduling*, p. 1.
[2] Cf. Domschke / Scholl (2008): *Grundlagen der Betriebswirtschaftslehre*, pp. 114.
[3] Graham et al. (1979): *Deterministic Sequencing and Scheduling.*

is a hybrid of flow shop and parallel machine problem, where both decisions have to be made simultaneously: 1. the sequence of jobs at each stage, 2. the assignment to one of the parallel machines. Further machine environments are job shop ($J||$) where jobs pass machines in different orders or open shop ($O||$) where jobs can be processed in any order. In addition to the sequence of processing steps, α may also define the number of machines. For example, ($F3||$) would describe a flow shop with 3 stages.

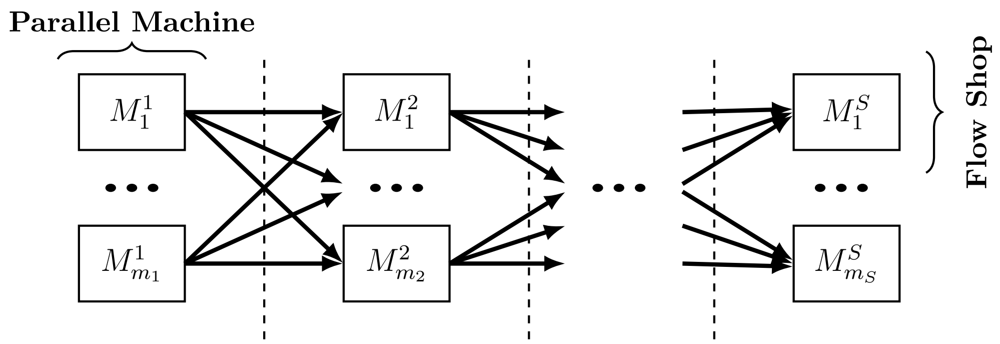


Figure 2.1: Machine environment in the hybrid flow shop[4]

Besides the machine environment, different **job characteristics** β determine the scheduling problem. There can be none, one or more properties for β. In this paper the following special features are discussed: different machine speeds ($P_{jk}(v_{jk})$), due dates (d_j), release dates (r_j), time-of-use energy prices (TOU), real time energy prices (RTP) and transportation efforts (t_{jk}).

Finally, the **objectives** γ significantly affect the decision problem. Different objectives can lead to various optimal schedules and can be contradictory in the multi-criteria case. In general, a distinction is made between capacity-oriented, time-oriented and cost-oriented objectives. In the following, total tardiness (TT), total energy costs (TEC), total costs (TC), makespan (C_{max}) and peak power (PP) will be considered.

If the problem is formally defined by the $\alpha|\beta|\gamma$ notation, complexity theory can be used to determine the size of the solution space and thus, how well the problem can be solved. Usually scheduling problems are complex combinatorial problems.[5] Although MIP formulations exist since the end of the 1950s and computing technology has developed significantly over the decades, exact solutions do mostly not exist for real problems.[6] To differentiate the problem complexity, problems are divided into $\mathcal{P}$ (solvable in polynomial time) and $\mathcal{NP}$ (solvable in non-polynomial time). All problems whose solution space size increases polynomially as a function of an input n belong to class $\mathcal{P}$. Thus, the complexity

[4]Based on Bruzzone et al. (2012): *Energy-aware scheduling*, p. 460.
[5]Cf. Eiselt / Sandblom (2004): *Decision analysis*, pp. 350.
[6]Cf. Pan (1997): *A study of integer programming formulations*, p. 33.

of each problem in $\mathcal{P}$ can be specified as $\mathcal{O}(n^{\delta})$ with $\delta > 0$. Otherwise, the problem can only be solved in non-polynomial time ($\mathcal{NP}$).[7]

Figure 2.2 illustrates the distinction between "polynomially solvable" and "NP-hard" for selected scheduling problems. The illustrated arrows show the connection between the individual problems, whereby the complexity increases with the direction of the arrows. This overview is limited to the two objectives makespan (maximum completion time) and total tardiness (sum of all delays), which are combined with energy considerations in the following. It can be seen that only special cases can actually be enumerated in polynomial computing time. For example, a flow shop with two machines can be optimally solved for the makespan criterion ($F2||C_{max}$) with Johnson's rule[8] but for more machines ($Fm||C_{max}$) it is NP-Hard.[9] The parallel machine problem on the other hand is already considered NP-hard with two machines for makespan ($P2||C_{max}$) and can only be solved efficiently for special cases like identical processing times ($p_j = p$).

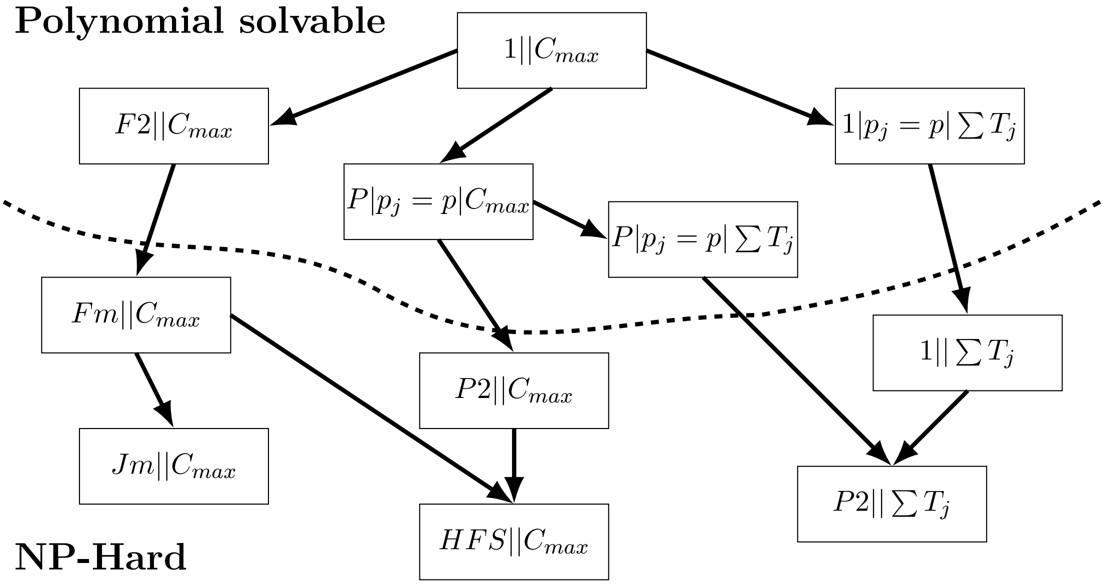


Figure 2.2: Complexity of different scheduling problems[10]

Total tardiness is a much more difficult criterion to optimize. Even for the simple case that only one machine is considered ($1||\sum T_j$), the problem is only in $\mathcal{P}$ if special cases such as $1|p_j = p|\sum T_j$ are analysed.[11] Overall, even simple parallel and flow shop problems are difficult to solve regardless of the objective. Consequently, HFS problems as a combination of these two problems are found almost exclusively in $\mathcal{NP}$. In addition, multiple objectives,

[7]For more information see e.g. Arora / Barak (2009): *Computational complexity: a modern approach.*
[8]Cf. Johnson (1954): *Optimal two- and three-stage production schedules.*
[9]Apart from special cases with three machines, which can be solved in polynomial time.
[10]Based on Pinedo (2012), p. 28. Data from Brucker / Knust (2009).
[11]Cf. Du / Leung (1990): *Minimizing total tardiness on one machine is NP-hard.*

variable production speeds, time-dependent energy costs and further complexity-driving properties are considered in the following. Thus, only very small problems can be solved exactly and a major task will be to develop efficient solution approaches.

Two further significant assumptions are made in this paper. In reality, schedules must be generated dynamically, as, for example, new jobs are opened or existing jobs are cancelled. In the following, however, a **static** observation is made and dynamic influences are not discussed in detail. It is assumed that the efficiency of the algorithms and the basic relationships between objectives and variables are not significantly influenced by regular data updates. Furthermore, real scheduling problems are often subject to uncertainties. For example, processing times may vary, machines can break down or prices are volatile. However, in practice, even simple processing times are often only estimated and the fluctuations or distributions are not known. The consideration of stochastic influences can therefore only be done under great uncertainty regarding the data. Hence, a **deterministic** consideration is made and stochastic uncertainties are not taken into account.

2.2 Basics of electricity trading

A large part of the scheduling literature deals with the assignment of machines to jobs. In reality, however, additional resources are often required. For example, workers may need to be assigned to machines,[12] material may be limited in availability[13] or budget has to be considered.[14] In this work, electrical energy is considered as an additional resource. The availability can be limited due to a maximum peak power, or costs can fluctuate over time. Electricity has special properties like being grid-bounded, flows according to resistance and can only be stored to a limited extent. This results in some special features in electricity trading, which will be briefly described here.

Due to the poor storability, demand and supply must always be balanced to ensure a stable network with a constant frequency (50 Hz in Germany). However, there are enormous fluctuations on both sides. Deviations in supply emerge due to power plant outages or varying feed-ins of renewable energy sources influenced by changing weather. With regard to demand, there are long-term (e.g. higher demand in winter than in summer) and short-term fluctuations (e.g. less at night than during the day). These differences lead to volatile energy prices as shown in Figure 2.3.

In scheduling, time-dependent energy prices are rarely taken into account. In fact,

[12]See e.g. Benavides / Ritt / Miralles (2014): *Flow shop scheduling with heterogeneous workers.*
[13]See e.g. Györgyi / Kis (2018): *Single machine with raw material.*
[14]See e.g. Zheng / Wang (2016): *Parallel machine scheduling problem with additional resource.*
[15]Data from EEX (2020): *Phelix Spot Market Prices.*

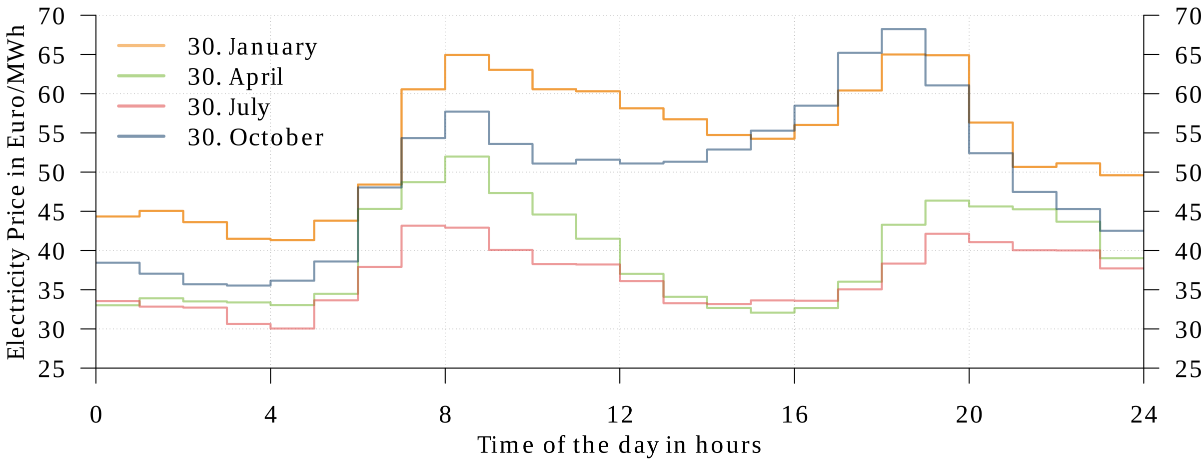


Figure 2.3: Spot market electricity prices in Germany on different days in 2019[15]

electricity costs are usually either neglected or only integrated by means of a fixed cost rate.[16] Therefore, this work will examine varying energy prices and is intended as a contribution to close the gap in research. In liberal, competitive energy markets, many different factors influence the price of electricity. Thus, predicting the price of electricity is a major challenge.[17] Nevertheless, significant trends can be seen and taken into account within production scheduling. As Figure 2.3 shows, the price of electricity is lower at night. During the day two peaks can be identified, one in the morning and one in the evening, both of which are mainly due to higher consumption in the private sector. Furthermore, differences can be observed with regard to the seasons. Prices are usually higher in winter, as considerably more electricity is needed for light and heat, and, at the same time, less electricity can be generated using solar energy.

When industrial companies trade directly on the exchange, energy-intensive operations should ideally be executed at night or in the afternoon. In addition, if long-term planning is possible, larger consumers should be used more intensively in summer. However, companies are subject to very different electricity contractual structures. There are basically two possibilities in electricity trading. Firstly, trading can take place on the stock exchange, whereby the producers feed into a large pool and the consumers draw from it. On the other hand, bilateral contracts can be negotiated. With both approaches, prices can be fixed in advance or calculated retrospectively based on the market situation.[18]

Another distinction made in the German energy market is between tariff customers and special contract customers. While private persons and smaller companies always purchase their energy from an electricity supplier according to tariff, companies with more

[16]Cf. Dong (2013): *Parallel machine scheduling*, p. 2240.
[17]Cf. Yang et al. (2020): *Electricity price forecasting*, p. 3.
[18]Cf. Bathurst / Weatherill / Strbac (2002): *Trading in short term energy markets*, p. 782.

than 30 MWh/a and at least two months with an average of 30 kW can conclude special contracts.[19] This means, on the one hand, that significantly lower fees have to be paid and, on the other, that bilateral contracts can be negotiated with the electricity supplier. Not even 10 TWh[20] of the almost 250 TWh[21] needed annually in the German industry are purchased according to tariffs. Consequently, almost all transactions are bilaterally negotiated. The exact details are usually not publicly known. Overall, more and more dynamic models are used which replace the fixed prices of the past.[22] Thus, companies are often paying variable prices. This demand response is important to give companies an incentive to reduce consumption when less renewable energy is available.[23]

Just as the individual contracts differ, various price models are considered in scheduling literature. For example, in load tracking the energy provider and the industrial consumer agree upon a target load curve and the company pays for deviations (called tracking errors). Minimizing these discrepancies by automatic scheduling leads to reduced energy cost.[24] The three most frequently examined models are **Time-of-Use** (TOU), **Real-Time Pricing** (RTP) or **Critical Peak Pricing** (CPP).[25] For TOU and RTP, time-dependent prices are paid per kWh. While TOU prices are fixed in advance for specific time windows (often on-, mid- and off-peak), RTP contracts require companies to pay according to the exchange price. Both approaches deal with the consumption charge for actual demand. TOU prices are analysed in Chapters 3 and 4 while RTP are integrated in Chapters 5 to 8.

In addition to the consumption charge, large consumers often pay a demand charge for the maximum peak power within the billing period (CPP). Thus, levelling the energy needs in order to lower the maximum load can reduce total energy costs enormously. However, the reduction of the peak power contradicts the intensification of production in times of low prices. This problem is dealt with in detail in Chapters 5 and 6.

2.3 Multi-criteria optimization

As the energy cost consideration is integrated into classic scheduling models and the objectives for ensuring production efficiency are simultaneously optimised, EAS requires usually multi-criteria approaches. For one-dimensional optimization problems, several solutions with the same objective function value may exist, but there is inevitably an

[19] Cf. Bundesamt für Justiz (2006): *Konzessionsabgabenverordnung §2 (7).*
[20] Cf. Rieseberg / Wörlen (2013): *Wachsende Strompreisunterschiede*, p. 6.
[21] Cf. BMWI (2019): *Gesamtausgabe der Energiedaten*, p. 21.
[22] Cf. Mathaba / Xia / Zhang (2014): *Electricity price forecast in industrial load scheduling*, p. 158.
[23] Cf. Anke et al. (2018): *Lastverschiebepotentiale in Dresden*, p. 3.
[24] See e.g. Hait / Artigues (2011a): *Scheduling with energy costs.*
[25] Cf. Bego / Li / Sun (2014): *Identification of reservation capacity in critical peak pricing*, p. 729.

optimal objective function value. For multi-criteria decisions, however, there are usually many different solutions with different objective values that come into question. These must first be found and then one of them has to be selected, which makes the problem much more complex than single-objective problems. Essential basics will be described in the following. In general, a multi-objective optimization problem can be defined as:[26]

$$\begin{aligned} &Minimize && \{\zeta_1(x), \zeta_2(x), ..., \zeta_k(x)\} \\ &Subject\ to && x \in S \end{aligned} \tag{2.1}$$

Here $\zeta(x)$ represents one of the k objective functions with $k \geq 2$. The set of possible decisions is represented by x and are located in the solution space S. This general formulation will be used in the following Chapters and applied to the specific problems under consideration.

For a better understanding, the following explanation focusses on problems with two objectives. The approaches can also be applied to k-dimensional problems. A simple example may serve as an illustration in Figure 2.4. A car trip can be optimized with respect to two objective functions $\zeta_1 - fuel\ consumption$ and $\zeta_2 - travel\ time$. If a higher speed is chosen, the travel time is reduced at the expense of fuel consumption. Additionally, other factors such as vehicle type, selected route or luggage may influence the objectives. These possibilities enable various decisions x which result in different objective function values.

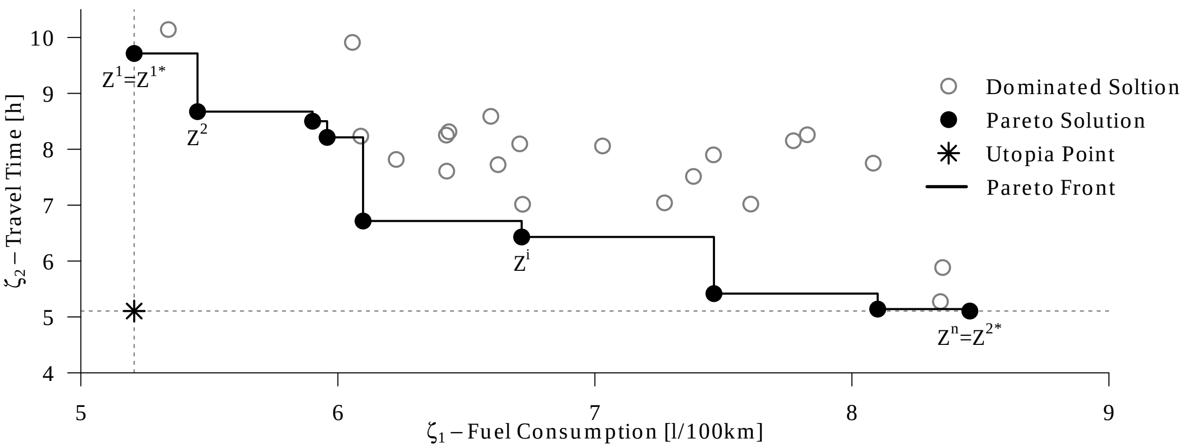


Figure 2.4: Pareto front for a bi-criteria example (car trip)

Although there are many different solutions, even in the case of multi-criteria problems, only a small number of solutions has to be considered by the decision maker. For example,

[26] Cf. Branke et al. (2008): *Multiobjective Optimization*, p. X.

a car trip with $7\,h$ driving time at a consumption of $6\,\frac{l}{100\,km}$ would never be chosen if a another route leads to a $6\,h$ drive with $6\,\frac{l}{100\,km}$. A solution $x^{'}$ that can be improved in one objective without worsening another criterion is called dominated. These solutions are not relevant for decision. From an optimization perspective, an efficient algorithm aims to find all solutions that are not dominated. Such **non-dominated solution**s (NDS) are also called pareto optimal. Generally, a solution is pareto optimal if no $\zeta_i(x^*)$ can be improved without deteriorating another $\zeta_j(x^*)$.[27]

Let Z^i be a pareto optimal solution. The n NDS for a bi-criterial problem can be sorted in ascending order using ζ_1. Then, all Z^i form the pareto front $\{Z^1, Z^2, ..., Z^n\}$ (see figure 2.4). The border solutions of the pareto front are called **lexicographic solutions**. Thereby, Z^{1*} contains the minimum of the first objective ζ_1^* and analogously Z^{2*} is an optimal solution for a single objective problem of the second criteria with ζ_2^*. The theoretically best solution as the intersection of ζ_1^* and ζ_2^* is called **Utopia Point**.[28]

All multi-criteria methods try to find pareto optimal solutions. A widespread differentiation of multi-criteria approaches is based on the time at which the decision maker interacts in the optimization:[29]:

1. **A priori:** If the decision-maker has a clear idea of the interdependencies between the objectives and can define clear preferences in advance, an optimal solution can be sought on this basis. Usually, the problem is thereby broken down into a one-dimensional problem. If, for example, the fastest route for the example in Figure 2.4 should be searched regardless of consumption, the lexicographic solution Z^{2*} corresponds to the optimal solution. This approach is particularly suitable if the objectives can be related to a single unit. This allows a **weighted sum** to be formed which can be optimized. This approach is also called blending and will be pursued in this paper in **Chapters 5, 7 and 8**. Therefore, all objectives are priced and thus a **global cost function** can be optimized. Other approaches for a priori problems include weighted min-max, weighted product or goal programming.[30]

2. **Interactive:** In interactive approaches the decision maker influences the optimization during the runtime. Thereby, the decision makers intervenes again and again in the algorithm. These approaches are also called progressive and will not be further analysed in this work.

3. **A posteriori:** While the previous approaches in principle only determine one

[27] Cf. Scholz (2018): *Multikriterielle Optimierung*, p. 172.
[28] Cf. Chircop / Zammit-Mangion (2013): *Epsilon-Constraint Based Methods*, p. 283.
[29] Cf. Emmerich / Deutz (2018): *A tutorial on multiobjective optimization*, p. 586.
[30] Cf. Marler / Arora (2004): *Survey of multi-objective optimization methods for engineering*, pp. 376.

solution, a posteriori approaches pursue all pareto optimal solutions. This enables the calculation of trade-offs between different decisions and with this a better basis for decision making. Furthermore, the dependencies of the variables and objectives can be better analysed. Consequently, in the following different a posteriori approaches are considered especially in **Chapters 3, 4 and 6**. At first, variants of the **epsilon constraint method** are implemented to generate optimal Pareto fronts.[31] Furthermore, **problem specific heuristic solution approaches** are developed. This allows to determine and explore the pareto front for larger problem instances. To evaluate the performances of the algorithms the well known **NSGAII** is used.[32]

[31] See e.g. Mavrotas (2009): *Effective implementation of the ε-constraint method*; Wang et al. (2018): *Bi-objective identical parallel machine scheduling*; Chircop / Zammit-Mangion (2013): *Epsilon-Constraint Based Methods.*

[32] Cf. Deb et al. (2002): *NSGA-II.*

3 Multi-objective hybrid flow shop scheduling with variable discrete production speed levels and time-of-use energy prices

Abstract

Energy costs play an important role in industrial production and are closely related to environmental concerns. As sustainability aspects are coming into focus in recent years, energy-oriented objectives are increasingly being taken into account in scheduling. At the same time, requirements for punctual delivery become more and more important in times of just-in-time delivery and highly networked supply chains. In this paper, a hybrid flow shop scheduling problem with variable discrete production speed levels is considered with the aim of minimizing both energy costs and total tardiness. Although lower speeds can reduce energy consumption, they also increase processing times, which counteract the objective of punctual delivery. Two new model formulations are presented and compared that take time-of-use energy prices into account. The influence of variable discrete production speed levels on energy costs, energy consumption and punctual delivery as well as the interdependencies between these objectives are analysed in a numerical case study.

Acknowledgement

Published Paper: S. Schulz / U. Buscher / L. Shen (2020): Multi-objective hybrid flow shop scheduling with variable discrete production speed levels and time-of-use energy prices. In: *Journal of Business Economics*, pp. 1 – 29.

3.1 Introduction

Hardly any product is as socially relevant as electrical energy. Many everyday objects only work with electricity while electrification is steadily increasing. In the course of Industry 4.0, industrial companies rely on automated processes using robots, driver-less transport systems or Auto ID technologies. In 2017, German industrial companies consumed 248.6 TWh of electrical energy and overall, the industrial sector is responsible for almost half of the total national electricity consumption.[1] The resulting CO_2 emissions amount to about one-fifth of total emissions.[2]

The great importance of energy not only leads to a great social interest in efficient and sustainable use, companies are also increasingly pressured to reduce their energy costs in the face of global competition. Furthermore, they can benefit from an environmentally oriented image. Consequently, energy costs are now being taken into account in many approaches of production planning and control and thus also in operative planning in the form of energy efficient scheduling (EES). Together with approaches to reduce emissions or waste and preserve resources, a completely new branch of green scheduling research has thus developed. A general overview about different approaches to consider energy consumption in scheduling is given by Biel / Glock (2016)[3] or Gahm et al. (2016)[4].

In this article we look at an extended flow shop problem, the HFS problem. In a flow shop problem, all jobs are processed in a multi-stage production in the same machine sequence, whereby only one machine is available at each stage. In contrast, in an HFS problem several machines are available on at least one stage. This allows, for example, to overcome step-related bottlenecks. The HFS problem can thus be seen as a generalization of a parallel machine problem and a flow shop problem, which can be found in many industrial processes such as electronics, paper, textile, pharmaceutical, and sheet metal industry.[5]

Among the EES approaches, variable production speeds are probably one of the most promising methods in order to significantly reduce energy consumption.[6] This article deals intensively with this topic. High savings potential exists, for example, in pumps which have high energy consumption in injection moulding plants or for water supply in paper mills. If a pump or fan works at 50% of the maximum volume flow, only 25% of the maximum pressure must be generated and the required power drops to 12.5%. This is based on the

[1] Cf. Ziesing (2018): *Energieverbrauch in Deutschland.*
[2] Cf. Dai et al. (2013): *Energy-efficient scheduling.*
[3] Biel / Glock (2016): *Energy-efficient production planning.*
[4] Gahm et al. (2016): *Energy-efficient scheduling.*
[5] Cf. Yu / Semeraro / Matta (2018): *A genetic algorithm for the hybrid flow shop.*
[6] Cf. Mecrow / Jack (2008): *Efficiency trends in electric machines and drives.*

affinity law, which states that the power consumption of a pump is proportional to the cube of the speed. Thus, even small changes in the flow rate can result in major energy savings.[7] Logically, the savings depend on the respective technical device. Even with industrial motors, enormous energy savings can be achieved through speed reductions. Besides lower energy consumption a reduction in energy costs can also be achieved through, inter alia, the exploitation of time-dependent energy prices, which are also considered in this work.

In addition to energy consumption and costs, other objectives are usually pursued. In recent years, several multi-criteria problems were published in EES. Commonly besides energy, utilization-oriented objectives as makespan are considered. In cases of strongly networked supply chains with just-in-time requirements, however, time criteria such as punctual delivery are playing an increasingly important role. Delayed production can lead to high contractual penalties and loss of confidence. Surprisingly, tardiness is rarely taken into account in EES. For that reason, this paper examines total tardiness and time depending energy costs as two separate objective functions using the ideas of multi-criteria optimization. To the best of our knowledge, this setting has not been addressed in HFS scheduling so far.

The article is structured as follows. Section 3.2 describes the current state of research. Subsequently, the problem is defined in section 3.3 and possible mathematical formulations are discussed. Approaches of multi-criteria decision-making are also explained here. A computational study follows in section 3.4. Finally, a short conclusion is given.

3.2 Related literature

In the following, the current state of research in EES including the fact that tardiness has hardly been taken into account so far is described. Due to the limited scope of the article, we will limit ourselves to machine and production scheduling in the following. However, it should be mentioned that the problem is basically similar to the multi-mode resource-constrained project scheduling problem (MMRCPSP). Mathematical problem formulations from this area can be found, for example, in Naber / Kolisch (2014)[8], Besikci / Bilge / Ulusoy (2015)[9] or Wauters et al. (2016)[10]. To the best of our knowledge, no MMRCPSP model can be directly applied to the problem considered here. The concept of multiple modes refers to the possibility to execute activities in different

[7] Cf. Lönnberg (2007): *Variable Speed Drives.*
[8] Naber / Kolisch (2014): *MIP models for resource-constrained project scheduling.*
[9] Besikci / Bilge / Ulusoy (2015): *Multi-project scheduling.*
[10] Wauters et al. (2016): *Multi-Project Scheduling.*

execution modes which allows to vary required time and resource consumption (e.g. energy). In addition to execution modes and variable speeds, the term different production rates is sometimes used. Since the term "variable speeds" is often applied in the field of machine scheduling and particularly for EES, we use it primarily in the article.

3.2.1 Energy efficient scheduling

There are various options to reduce energy costs in scheduling, whereby the possibilities also depend on the respective production processes. Basically, either energy consumption is reduced directly through intelligent planning (section 3.2.1) or pricing mechanisms are exploited to reduce expenses while energy consumption stays at the same level (section 3.2.1). An overview is shown in Fig. 3.1. The concepts a) to f) are explained in detail in the following. The framed approaches in Fig. 3.1 will be taken into account in the considered problem. In addition to the approaches listed here, there are a few very specific contributions, which are not taken into account. For example, Nolde / Morari (2010)[11] as well as Hait / Artigues (2011b)[12] examine load tracking scheduling in a steel plant. Energy provider and consumer may agree upon a target load curve. For deviations the company has to pay (called tracking errors). Also Modos / Sucha / Hanzalek (2017)[13] examine this problem.

Furthermore, this paper concentrates on electrical energy. The EES literature also contains contributions that deal with heat, cold, water or emissions. A good overview about all research trends in EES is given by Gahm et al. (2016)[14]. They also propose a classification framework. This contribution can be classified in this respect by $ECS, PS|TOU|FLX$ indicating:

- Energy Coverage: External conversion system (ECS), production system (PS),
- Energy Supply: Price driven demand response by time-of-use prices (TOU),
- Energy Demand: Flexible (FLX) processing energy demand.

Utilisation of market mechanisms with constant consumption

While private consumers are bound to fixed tariffs, companies with an annual consumption of 10 MWh or more can negotiate special contracts with the respective energy supplier. Most of these bilateral contracts are not publicly available. Nevertheless, it is known that

[11] Nolde / Morari (2010): *Electrical load tracking.*
[12] Hait / Artigues (2011b): *Electrical load tracking scheduling.*
[13] Modos / Sucha / Hanzalek (2017): *Algorithms for robust production scheduling.*
[14] Gahm et al. (2016): *Energy-efficient scheduling.*

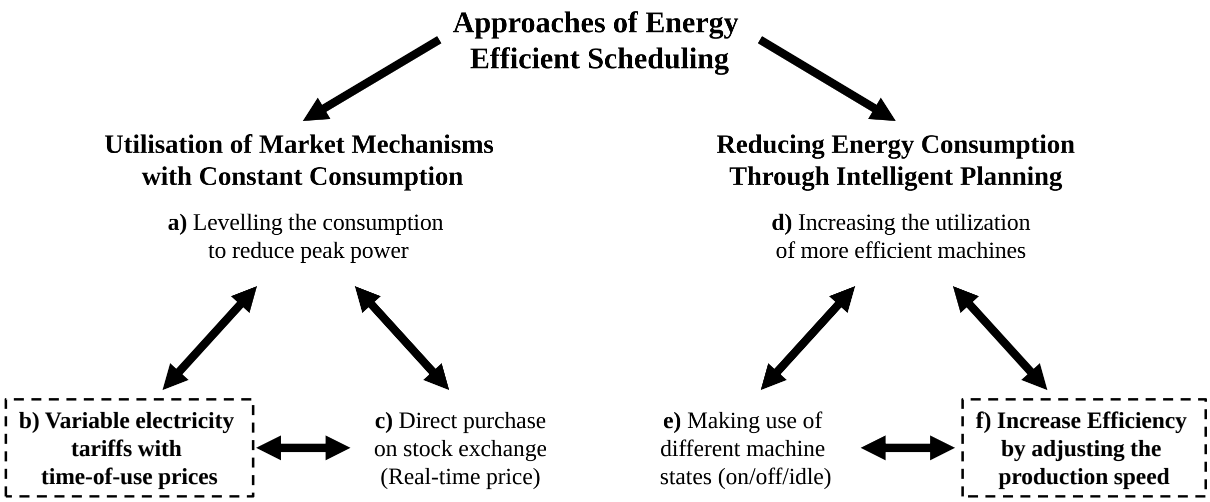


Figure 3.1: Overview about different EES approaches

companies usually have to pay an electricity charge per kWh consumed and a so-called demand charge for the respective maximum peak power within the billing period.[15]

a) Peak Power Reduction

To reduce costs due to peak power consumption, peaks must be reduced or entirely avoided. While the peak power fee normally covers at least a time period of several weeks, scheduling is primarily dedicated to shorter production periods. Therefore, the direct integration of the demand charge in a scheduling problem is not straightforward possible. However, since the demand charge per kW is 200 to 400 times higher than the electricity price per kWh[16], reducing the energy peak can be fairly lucrative. The peak power is often considered as a constraint and set to a historical value which must not be exceeded.[17] A parametric optimization is also possible by varying the respective upper bound. A direct minimization within the objective function is done for example by Nagasawa / Ikeda / Irohara (2015)[18] who present a simulation approach for a flow shop problem with random processing times. Strong fluctuations in energy consumption does not only lead to high peak power costs. Constant energy consumption is also important for internal power generators or converters. Rager / Gahm / Denz (2014)[19] publish an approach which aims to minimize the sum of the squared deviations of each load from the expected average energy consumption to improve the performance of an energy conversion system.

[15]Cf. Bego / Li / Sun (2014): *Identification of reservation capacity in critical peak pricing.*
[16]Cf. Nghiem et al. (2011): *Green scheduling.*
[17]See e.g. Bruzzone et al. (2012): *Energy-aware scheduling*; Xu / Weng / Fujimura (2014): *Energy-Efficient Scheduling*; Schulz (2018): *A Multi-criteria MILP Formulation.*
[18]Nagasawa / Ikeda / Irohara (2015): *Robust flow shop scheduling.*
[19]Rager / Gahm / Denz (2014): *Energy-oriented scheduling.*

b) Time-of-Use Prices

Electricity charges may depend on the time the energy is used, which requires consumption to be postponed in times of lower prices. These shifts are in contrast to the levelling due to the demand charge. One of the first contributions dealing with time-of-use tariffs (TOU) in production scheduling was published by Nilsson / Söderström (1993)[20]. Castro / Harjunkoski / Grossmann (2009)[21] present a continuous-time scheduling formulation for an EES model considering two different TOU energy prices. As a result, the energy costs can be reduced by around 20% by moving consumption from on-peak to off-peak times. Che / Zhang / Wu (2017)[22] consider an unrelated parallel machine problem and suggest a constructive heuristic to minimize energy costs and makespan simultaneously.

c) Real-Time Prices

While TOU tariffs usually have two (on-/off-peak) or three (on-/mid-/off-peak) different time-depending price levels, Ding et al. (2016)[23] analyse the influence of frequently fluctuating TOU prices in an unrelated parallel machine scheduling problem. Such problems with hourly fluctuating energy prices are often referred to as real-time prices (RTP). Energy-intensive companies with annual consumption of more than 100 MWh may purchase electricity directly from the stock exchange. This means, they pay the RTP for quantities that are not hedged by long-term derivatives. These prices fluctuate at the EEX (European Energy Exchange) every 15 minutes, which increases the computational effort for optimization enormously. Mitra et al. (2012)[24] describes mixed-integer program (MIP) formulations for the production planning of a week with hourly fluctuating energy prices. In Küster et al. (2013)[25], a complex production process with RTP is visualised as a bipartite graph. The authors then present a simulation and optimization approach. They explain that in times of lower electricity prices more renewable energies are fed into the grid and thus not only costs are saved but the environment can also be protected.

[20] Nilsson / Söderström (1993): *Industrial applications of production planning.*
[21] Castro / Harjunkoski / Grossmann (2009): *New Continuous-Time Scheduling.*
[22] Che / Zhang / Wu (2017): *Energy-conscious unrelated parallel machine scheduling.*
[23] Ding et al. (2016): *Parallel machine scheduling under time-of-use electricity prices.*
[24] Mitra et al. (2012): *Optimal production planning.*
[25] Küster et al. (2013): *Distributed evolutionary optimisation.*

Reduce energy consumption through intelligent planning

Reducing energy costs by making use of market mechanism does not influence the energy consumption and can even have a negative impact on CO_2 emissions and other environmental factors. Zhang et al. (2014)[26] discuss the correlation between utilization of TOU tariffs and CO_2 emissions. They show that in times of low electricity prices, emissions are on average higher than those in on-peak periods. Interestingly, this contradicts the statement of Küster et al. (2013)[27]. Overall, both points of view can be understood and it depends largely on the energy market under consideration whether low prices are accompanied by lower emissions. For example, electricity prices are usually low at night. Since only few renewable energy sources can be used at night, this electricity is often generated by conventional power plants such as coal. The extent to which renewable energies are freely traded on the market or are subsidised must also be taken into account. In any case, from an environmental point of view, it may be beneficial if scheduling also reduces energy consumption and considers environmental impacts as well. For this purpose, three different approaches are particularly considered in literature as can be seen in Fig. 3.1.

d) Increase utilization of more efficient machines
In heterogeneous production environments with parallel machines which have different energy consumptions and processing times for the same task, consumption can be reduced through higher utilization of more efficient machines. Ji / Wang / Lee (2013)[28] consider a uniform parallel machine problem in which machines with higher resource consumption also work faster. The authors present an MIP as well as a particle swarm heuristic to optimize resource consumption for a given maximum makespan, whereby the term resource is not limited to energy. A particle swarm optimization is also used by Nilakantan / Huang / Ponnambalam (2015)[29] to minimize makespan and energy consumption in a robotic assembly line system with differently efficient robots. Schulz / Neufeld / Buscher (2019)[30] propose an iterated local search algorithm to optimize three different objectives (makespan, energy costs, peak power) in a heterogeneous hybrid flow shop problem. In this work, however, we focus on identical parallel machines.

e) Making use of different machine states
Various EES contributions take different machine states into account. The basic idea is to

[26] Zhang et al. (2014): *Energy-conscious flow shop scheduling.*
[27] Küster et al. (2013): *Distributed evolutionary optimisation.*
[28] Ji / Wang / Lee (2013): *Minimizing resource consumption.*
[29] Nilakantan / Huang / Ponnambalam (2015): *An investigation on minimizing cycle time.*
[30] Schulz / Neufeld / Buscher (2019): *Comprehensive energy-aware hybrid flow shop.*

optimize idle and standby times as well as making intelligent shut down decisions. Liu et al. (2014a)[31] analyse a job shop problem with three different processing levels (idle, runtime, cutting), whereby the energy consumption is a deterministic value and can only be reduced by minimizing the non-processing time. Also in the HFS problem considered by Dai et al. (2013)[32] the basic idea is to minimize the idle energy consumption. If it is possible to shut down machines, the amount of possible energy savings increases. This concept is examined for example by Mashaei / Lennartson (2013)[33]. They minimize energy consumption in a flow shop problem for a given cycle time by weighing between switching off and idling in times of no production. Thereby, switching on and off leads to higher energy consumption than short idle times, but is advantageous during longer periods of standstill. Li et al. (2018)[34] assume that setup energy is required after idle times and the objective is to minimize makespan and total energy consumption in an HFS. In Wu / Sun (2018)[35] on/off decisions are not only used to reduce energy consumption in a flexible job shop problem, but the total number of turning-on/off machines is minimized as a third objective besides makespan and total energy consumption.

f) Variable production speed

Most EES articles consider discrete constant energy consumption depending on the machine state, which is only an approximation of actual energy requirements. Therefore, often data from energy audits are used, which provide average values.[36] Real production processes are subjected to various factors and uncertainties.[37] Only a few authors consider realistic models for energy consumption. For example, Yan et al. (2016)[38] look at different cutting machines within an HFS and propose a multi-level optimization approach to minimize makespan and energy consumption when taking into account different cutting speeds. Thus, in addition to random influences such as machine ageing, environmental effects or material parameters, energy consumption can also be directly influenced by variable production speed. That idea is used in different EES approaches to reduce energy costs, energy consumption or emissions and is also considered in this work.

For example, Fang / Lin (2013)[39] propose an MIP formulation for flow shop problems. A further approach with processing time depending energy consumption is published by

[31] Liu et al. (2014a): *Minimising total energy consumption.*
[32] Dai et al. (2013): *Energy-efficient scheduling.*
[33] Mashaei / Lennartson (2013): *Energy Reduction in a Pallet-Constrained Flow Shop.*
[34] Li et al. (2018): *Efficient multi-objective optimization.*
[35] Wu / Sun (2018): *A green scheduling algorithm.*
[36] See e.g. Abdelaziz / Saidur / Mekhilef (2011): *A review on energy saving strategies.*
[37] See e.g. Le / Pang (2013): *Fast reactive scheduling.*
[38] Yan et al. (2016): *Energy-efficient flexible flow shop scheduling.*
[39] Fang / Lin (2013): *Parallel-machine scheduling.*

Zanoni / Bettoni / Glock (2014)[40]. They consider a two-machine problem with three storages to minimize the total costs of production, storage and energy. Liu / Zhao / Xu (2012)[41] examine the interdependencies between process efficiency and energy consumption of an electroplating unit in a hoist scheduling problem. A stochastic problem with non-linear energy cost function depending on variable production quantity is formulated by Tang / Che / Liu (2012)[42]. In Hait / Artigues (2011a)[43] the task duration depends on the given power to a furnace. A model and constructive heuristic for a bi-objective two stage flow shop with three different speed levels is described in Mansouri / Aktas (2016)[44]. Lei / Zheng / Guo (2017)[45] examine a flexible job shop scheduling problem with variable discrete production speeds. To minimize total energy consumption and workload balance they propose a shuffled frog-leaping algorithm. In a later work Lei / Gao / Zheng (2018)[46] present a novel teaching-learning algorithm to minimize total energy consumption and total tardiness in a hybrid flow shop scheduling problem. The problem is basically similar to the one we are investigating here. However, no TOU prices are considered, so the decision problem is limited to semi-active schedules. In the following a few more EES articles are described, which consider tardiness as objective.

3.2.2 Multi-criteria EES with total tardiness

EES approaches often consider further objectives besides energy demand or costs. However, this is usually limited to makespan. In our opinion, it is desirable to include energy costs and tardiness in an approach, especially since it is hardly been done so far. To the best of our knowledge this is the first time that both objectives are considered in an HFS. Besides HFS there are a few publications that investigate total tardiness in multi-criteria EES.

Artigues / Lopez / Hait (2013)[47] minimize energy and power overrun costs in a parallel machine problem, and use maximum tardiness as decision criterion for the scheduling in the first step of a two-phase solution approach. Also Liu / Lee / Wang (2016)[48] consider a parallel machine problem. They propose a branch and bound algorithm to minimize the resource consumption with maximum tardiness as a constraint. In Liu /

[40] Zanoni / Bettoni / Glock (2014): *Energy implications.*
[41] Liu / Zhao / Xu (2012): *Integration of electroplating process design and operation.*
[42] Tang / Che / Liu (2012): *A stochastic production planning problem.*
[43] Hait / Artigues (2011a): *Scheduling with energy costs.*
[44] Mansouri / Aktas (2016): *Minimizing energy consumption and makespan.*
[45] Lei / Zheng / Guo (2017): *A shuffled frog-leaping algorithm.*
[46] Lei / Gao / Zheng (2018): *Teaching-learning-based optimization algorithm.*
[47] Artigues / Lopez / Hait (2013): *The energy scheduling problem.*
[48] Liu / Lee / Wang (2016): *Resource consumption minimization.*

Zhou / Yang (2017)[49] a fuzzy flow shop problem is described with tardiness costs and energy costs summarized in one objective function. A similar cost function is used by Le / Pang (2013)[50], whereby uncertainties in energy consumption are taken into account. Tardiness is considered in the four publications mentioned, but does not appear as an independent aim in the objective function.

A genetic algorithm for the multi-objective job shop problem with minimization of energy consumption and total weighted tardiness is introduced by Zhang / Chiong (2016)[51]. The same objectives are considered by Wang et al. (2016)[52] in a batch scheduling problem under energy consumption uncertainties. In Che et al. (2017)[53] energy consumption and maximum tardiness are minimized in a single machine problem with power down options. The last three approaches mentioned describe multi-criteria approaches with energy consumption and tardiness as objective functions. However, variable energy prices or discrete speeds are not taken into account.

3.3 Problem description

In this section we will define the considered problem in detail. To analyse the interdependencies between total tardiness and energy costs a multi-objective MIP formulation is developed. An overview on notation with indices, parameters and variables is shown in Table 3.1.

3.3.1 Assumptions

We consider an HFS problem where n jobs go through m stages (with $m \geq 2$) following the same processing sequence (flow shop). Each stage k consists of identical parallel machines i ($i \in \{1, \ldots, \mu_k\}$). Thereby, each machine at a stage has the same technical requirements but can differ in terms of speed. A typical machine layout can be seen in Fig. 3.2.

For practical implementation of variable motor speeds, electronic voltage converters are connected upstream of an electric motor. As a result, speed, torque, as well as the resulting power can be varied arbitrarily.[54] Such technologies are already used in various areas such as building management.[55] Thus, an infinite number of speeds is theoretically possible.

[49]Liu / Zhou / Yang (2017): *Minimizing energy consumption and tardiness penalty.*
[50]Le / Pang (2013): *Fast reactive scheduling.*
[51]Zhang / Chiong (2016): *Solving the energy-efficient job shop.*
[52]Wang et al. (2016): *Batch scheduling.*
[53]Che et al. (2017): *Energy-efficient bi-objective single-machine scheduling.*
[54]Cf. Abdelaziz / Saidur / Mekhilef (2011): *A review on energy saving strategies.*
[55]See e.g. Saidur (2009): *Analysis in Malaysian office buildings.*

Indices	
i	Machine in $\mathcal{M}_k = \{1, ..., \mu_k\}$
j	Job in $\mathcal{J} = \{1, ..., n\}$
k	Production stage or task in $\mathcal{S} = \{1, ..., m\}$
l	Level of speed reduction as additional processing time in $\mathcal{V} = \{0, \ldots, o\}$
t	Discrete time-interval in $\mathcal{T} = \{1, ..., \tau\}$
Parameters	
ed_{jk}	Maximum energy demand at maximum speed of task k of job j
es_{jkl}	Energy consumption of task k of job j at the speed reduction l
es^*_{jkl}	Relative energy saving if speed is reduced from $(l-1)$ to l
D_j	Due date of job j
ec^t	Electricity cost during time period t
p_{jk}	Minimum processing time of task k of job j
Decision Variables	
$c_{jk} \in \mathbb{N}^+$	Completion time of task k of job j
$e^t_{jk} \in \mathbb{R}^+$	Energy consumption of task k of job j at time period t
$EC_{jk} \in \mathbb{R}^+$	Energy costs of task k of job j
$EP_{jk} \in \mathbb{R}^+$	Energy consumption of task k of job j
$g_{jkl} \in \{0,1\}$	The speed reduction l of task k of job j is set as individual variable
$g^*_{jkl} \in \{0,1\}$	The speed reduction l of task k of job j is set as special ordered set
P_{jk}	Actual processing time of task k of job j
$s_{jk} \in \mathbb{N}^+$	Start time of task k of job j
$T_j \in \mathbb{N}^+$	Tardiness of job j
$x^t_{jk} \in \{0,1\}$	Task k of job j is performed at time t
$z^t_{jk} \in \{0,1\}$	Execution of task k of job j starts at time t

Table 3.1: Used notation for the MIP formulation in section 3.3

However, this cannot be formulated in an MIP, as the solution space would also become infinitely large. Consequently, a finite and discrete set of speed levels $\mathcal{V} = \{0, \ldots, o\}$ is available for each machine in order to control processing speed. The speed is determined individually for each job and remains constant during the processing of a job.

Since we assume discrete integer production times, reducing the speed by one step leads to one additional processing time unit. Thus, the variable l can simultaneously represent

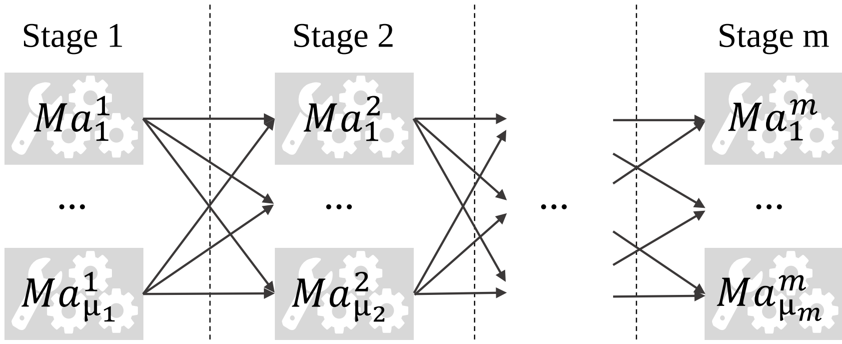


Figure 3.2: General machine layout in a HFS problem

additional processing time and a reduction in speed. To avoid ambiguity, l is referred to as the level of speed reduction in the following. A higher l increases the processing time but decreases energy consumption. The operation of task k of job j needs at least a baseline processing time p_{jk} which corresponds to maximum speed as well as maximum energy demand. To reduce the maximum speed by one step, l is set to 1. This increases the processing time by one unit. The resulting actual processing time is called P_{jk} and would equal $p_{jk} + l$ with $l = 1$.

Through speed variations the energy consumption can be influenced. The energy cost EC_{jk} of job j at stage k is determined by the time-dependent energy prices ec^t and the speed-dependent consumption EP_{jk}. Thereby, energy consumption EP_{jk} does not depend linearly on the speed reduction, but the interdependencies are much more complex. Often cubic relationships are assumed on the basis of the affinity laws which are already mentioned in the introduction. In our model the maximum energy consumption at highest speed is ed_{jk}. Based on that value and the actual processing time P_{jk}, the following relationship is assumed:

$$EP_{jk} = \left[1 + 0.6 \cdot \left(\frac{P_{jk}}{p_{jk}} - 1\right)^2 - 1.4 \cdot \left(\frac{P_{jk}}{p_{jk}} - 1\right)\right] \cdot ed_{jk} \cdot \frac{p_{jk}}{P_{jk}} \qquad \forall\ j, k. \tag{3.1}$$

This equation is based on the work of Almeida / Ferreira / Both (2005)[56] and represents an electric motor for a pump with variable speed drive. A possible practical application could be an injection moulding machine. The minimum energy consumption is achieved at a ratio of P_{jk} over p_{jk} of 2.09, which means that processing speed reduction up to 52.15% may reduce total energy consumption. This value meets practical experiences. Because of excessive wear and the inefficient working area, electric engines lose efficiency if

[56] Almeida / Ferreira / Both (2005): *Application of variable-speed drives.*

driven below 50% of rated load.[57] For that reason, in our test instances speed reductions are only permitted up to 50%. Since equation (3.1) is quadratic, it is helpful to linearise the expression in the course of MIP modelling.

Energy costs can be reduced on the one hand by reducing the production speed and on the other hand by shifting the work to off-peak time periods with lower energy prices. If only energy costs were minimized, jobs would be processed at very low speeds and mainly times of low electricity prices would be used for production. This could increase cycle times enormously. We assume, that each job j has a due date D_j. If the completion time exceeds the due date the difference is called tardiness T_j (see equation (3.11)). In order to obtain a time-efficient schedule despite the energy cost optimization, we minimize the total tardiness of all jobs as a second objective. On the basis of these assumptions, we consider the following two objectives (3.2) and (3.3).

$$\textbf{Minimize:}\ \text{I. Total Electricity Costs:}\quad TEC = \sum_{k=1}^{m}\sum_{j=1}^{n} EC_{jk} \tag{3.2}$$

$$\text{II. Total Tardiness:}\quad TT = \sum_{j=1}^{n} T_j \tag{3.3}$$

Besides the mentioned properties, the following general assumptions are made for the HFS problem:

- All jobs and machines are available at time zero (no release dates).
- Each machine can process at most one job at a time.
- Each job can be processed by at most one machine at a time.
- Once a task has been started, no interruption is allowed.
- There are infinite buffers between stages.
- Set-up effort and transportation times are neglected.

3.3.2 Time-indexed model formulation

The described problem can be formulated as an MIP. Since we consider time depending energy cost, the model is set up using time-indexed variables. The planning horizon is subdivided into τ time periods t ($t \in \{1, \dots, \tau\}$). We introduce two binary decision variables. The binary x_{jk}^t is equal to one, if job j is processed in t at stage k. Similarly, the binary z_{jk}^t is one, if job j starts at stage k in period t. Based on that two decision

[57] See e.g. Saidur et al. (2009): *Energy and emission analysis.*

variables, the basic constraints can be formulated as follows.

$$\sum_{j=1}^{n} x_{jk}^{t} \leq \mu_k \qquad \forall\, k, t \tag{3.4}$$

$$\sum_{t=1}^{\tau} x_{jk}^{t} = P_{jk} \qquad \forall\, j, k \tag{3.5}$$

$$\sum_{t=1}^{\tau} z_{jk}^{t} = 1 \qquad \forall\, j, k \tag{3.6}$$

$$x_{jk}^{1} = z_{jk}^{1} \qquad \forall\, j, k \tag{3.7}$$

$$x_{jk}^{t} - x_{jk}^{t-1} \leq z_{jk}^{t} \qquad \forall\, j, k, t > 1 \tag{3.8}$$

$$s_{jk} = \sum_{t=1}^{\tau} \left(z_{jk}^{t} \cdot t \right) \qquad \forall\, j, k \tag{3.9}$$

$$c_{jk} = s_{jk} + P_{jk} - 1 \qquad \forall\, j, k \tag{3.10}$$

$$T_j = \max \{0, c_{jm} - D_j\} \qquad \forall\, j \tag{3.11}$$

$$s_{jk} - c_{jk-1} \geq 1 \qquad \forall\, j, k > 1 \tag{3.12}$$

Constraint (3.4) ensures that the number of parallel processes in each stage complies with the number of parallel machines. Thus, no machine can be assigned multiple jobs at the same time. By introducing (3.5) each job is scheduled for the entire processing time needed. Furthermore, with (3.6) a job can start only once. Expressions (3.7) and (3.8) connect both binary variables. These three equations operate together to ensure that tasks cannot be interrupted.

In addition, we introduce two further dependent decision variables by equations (3.9) and (3.10). The integer s_{jk} is the start time period of task k of job j and the integer c_{jk} represents the completion time of a job j at stage k. Both variables depend on z_{jk}^{t} as well as the chosen speed and do not necessarily have to be introduced. However, it simplifies the formulation of the model and accelerate the optimization by means of solver. On the basis of c_{jk} it is then possible to calculate the tardiness T_j of job j with (3.11). Equation (3.12) ensures that a job can be processed at a stage only if the previous step is completed.

Due to the quadratic function (3.1) and the dependence of the energy consumption on the processing speed, neither direct proportionality nor additivity is given. For linearisation, we make two modifications. Firstly, function (3.1) is calculated for all possible levels of speed reduction $l \in \{0, \ldots, o\}$ and the result is saved as parameter es_{jkl}. Secondly, the interdependencies of decision variables are dissolved by introducing a further binary auxiliary variable g_{jkl}. Thereby, the binary g_{jkl} is equal to 1, if job j is processed at stage

k at level of speed reduction l. The following constraints have to be added.

$$\sum_{l=0}^{o} g_{jkl} = 1 \qquad \forall\, j,k \tag{3.13}$$

$$P_{jk} = p_{jk} + \sum_{l=0}^{o} (g_{jkl} \cdot l) \qquad \forall\, j,k \tag{3.14}$$

$$e_{jk}^{t} \geq \left[\sum_{l=0}^{o} (g_{jkl} \cdot es_{jkl})\right] - ed_{jk} \cdot \left(1 - x_{jk}^{t}\right) \qquad \forall\, j,k,t \tag{3.15}$$

$$e_{jk}^{t} \geq 0 \qquad \forall\, j,k,t \tag{3.16}$$

$$EC_{jk} = \sum_{t=1}^{\tau} \left(e_{jk}^{t} \cdot ec^{t}\right) \qquad \forall\, j,k \tag{3.17}$$

Constraint (3.13) assigns exactly one level of speed reduction l to each job j at each stage k. Then P_{jk} can be calculated by (3.14). The respective consumption e_{jk}^{t} is calculated in equation (3.15) depending on the additional processing time. Since ed_{jk} is always greater than or equal to the actual energy consumption, it works similar to a Big M formulation and there is only a positive value if actual production takes place. Furthermore, (3.16) ensures non-negative energy consumption if not manufactured ($x_{jk}^{t} = 0$). Finally, (3.17) sums up the energy costs for each job at each stage.

3.3.3 Model improvements

The formulation described in section 3.3.2 is complete and can be solved by solver. However, the formulation can still be improved. The number of variables g_{jkl} is very high. The speed is selected by one single binary without any connection between individual speed levels. If special ordered sets are used instead, the branch and bound procedure can be significantly accelerated.[58] In doing so, equations (3.13) to (3.15) are replaced by (3.18) to (3.20).

$$g_{jkl-1}^{*} \geq g_{jkl}^{*} \qquad \forall\, j,k,l : l > 1 \tag{3.18}$$

$$P_{jk} = p_{jk} + \sum_{l=1}^{o} g_{jkl}^{*} \qquad \forall\, j,k \tag{3.19}$$

$$e_{jk}^{t} \geq ed_{jk} \cdot \left[x_{jk}^{t} - \sum_{l=1}^{o} \left(g_{jkl}^{*} \cdot es_{jkl}^{*}\right)\right] \qquad \forall\, j,k,t \tag{3.20}$$

The basic idea is that the processing time is stepwise increased by g_{jkl}^{*}. If time is increased by a certain level Φ all previous levels must also be activated. In the model this

[58]For more information see e.g. Beale / Forrest (1976): *Global optimization using special ordered sets.*

means that not only $g^*_{jk\Phi} = 1$ but all $g^*_{jkl} = 1$ with $l \leq \Phi$. This relationship is established by constraint (3.18). Then, the additional processing time can be calculated by the sum of all g^*_{jkl} in (3.19). Finally, the time depending energy consumption must be calculated by equation (3.20). Here, es^*_{jkl} is the relative percentage energy saving for job j at stage k if the level of speed reduction is changed from $l-1$ to l.

With the reformulation described above, the computation time can be reduced by up to 50% for certain test instances. Moreover, some other approaches have been tested to speed up the solution finding. Various additional constraints were integrated, which unfortunately did not lead to any significant improvement. In addition, optimization could be probably accelerated if starting solutions are generated. But finding initial solutions does not seem to be a problem here. For example, an initial solution was created with the earliest due date rule. Thereby the jobs are scheduled based on their due dates (job with the smallest D_j is scheduled first and so on). Interestingly, computation time was even significantly extended by using initial solutions.

Since the described model cannot be further improved directly, other formulations are investigated. The calculation of energy costs is quite complex due to the fluctuating energy prices and the time dependency. Furthermore, it has to be repeated frequently which requires a lot of computing time. One alternative is to calculate all possible energy costs depending on job, stage, speed and starting time in advance. Combining this idea with a sequence-dependent formulation leads to another MIP, which is described in the following.

3.3.4 Sequence-dependent model formulation

The new model is based on the parameter tec^t_{jkl} which defines the energy costs for a job j at stage k, if processing starts in time period t at level of speed reduction l. This parameter is calculated in a pre-process. Since the determination takes much less than a second for the considered problem sizes this calculation step is not analysed separately in the following. Rather, we focus on the resulting novel model formulation. Just like tec^t_{jkl}, the processing time P_{jkl} depends on l. Thus, speed respectively extension of the processing time is considered as an index in the following. The modified decision variables are shown in Table 3.2. Further notation is similar to Table 3.1.

$x_{jj'k} \in \{0,1\}$	Job j starts after j' at stage k.
$y_{jki} \in \{0,1\}$	Job j is executed by machine i at stage k
$z^t_{jkl} \in \{0,1\}$	Execution of task k of job j starts at time t with speed l

Table 3.2: New decision variables for the modified model

While the objective functions remain unchanged, constraints are mainly modified as can be seen in equation (3.21) to (3.30).

$$x_{jj'k} + x_{j'jk} \geq y_{jki} + y_{j'ki} - 1 \qquad \forall\, i, j, k, j' \neq j \tag{3.21}$$

$$\sum_{i=1}^{\mu_k} y_{jki} = 1 \qquad \forall\, j, k \tag{3.22}$$

$$\sum_{l=0}^{o} \sum_{t=1}^{\tau} z_{jkl}^{t} = 1 \qquad \forall\, j, k \tag{3.23}$$

$$s_{jk} = \sum_{l=0}^{o} \sum_{t=1}^{\tau} z_{jkl}^{t} \cdot t \qquad \forall\, j, k, \tag{3.24}$$

$$c_{jk} = s_{jk} + \sum_{l=0}^{o} \sum_{t=1}^{\tau} z_{jkl}^{t} \cdot P_{jkl} - 1 \qquad \forall\, j, k \tag{3.25}$$

$$s_{jk} - c_{jk-1} \geq 1 \qquad \forall\, j, k > 1 \tag{3.26}$$

$$c_{j'k} \leq s_{jk} + (1 - x_{jj'k}) \cdot \tau - 1 \qquad \forall\, j, k, j' \neq j \tag{3.27}$$

$$c_{jm} \leq \tau \qquad \forall\, j \tag{3.28}$$

$$T_j = \max\{0, c_{jm} - D_j\} \qquad \forall\, j \tag{3.29}$$

$$EC_{jk} = \sum_{l=0}^{o} \sum_{t=1}^{\tau} z_{jkl}^{t} \cdot tec_{jkl}^{t} \qquad \forall\, j, k \tag{3.30}$$

Equation (3.21) specifies that each job j must be either predecessor or successor of job j' if both jobs are processed on the same machine. With (3.22) each job is allocated to a machine. Constraint (3.23) guarantees that each job starts at each stage exactly once with a certain speed. In (3.24) and (3.25) start and completion time period are defined. On that basis with (3.26) a job can only starts if the previous stage is finished. Similarly, a job can start at a machine only if all jobs which are scheduled earlier are finished. For this purpose, constraint (3.27) prevents overlapping of jobs allocated to the same machine. Of course, the completion time of each job must not exceed τ which leads to (3.28). In the time-indexed model equation (3.5) ensures that the complete processing time is within the observation period. Similar to the first model the tardiness of each job is calculated in (3.29). Finally, depending on the data of tec_{jkl}^{t}, (3.30) calculates the energy costs for each job at each stage.

The performance of this approach in comparison with the model presented first will be evaluated in a computational study in section 3.4. Beforehand, the problem of multi-criteria decision making will be discussed briefly.

3.3.5 From lexicographic to eps-constraint method

Due to the different objective functions (3.2) and (3.3), the described problem can not be solved directly by a solver. Basically, Branch and Cut algorithms are used to optimize a single objective function. However, if there are several objectives, one possibility is to combine them into a single function. For example, one could try to monetize the values and thus obtain a global cost function. In the considered problem, unfortunately, a weighted-sum approach is not possible to combine both criteria in one function. Delays in particular are difficult to monetise. Contractual penalties often occur, but in addition, trust and goodwill losses must also be taken into account.

Another possible approach is lexicographical optimization. Therefore, the objectives are put into a certain order and then the criteria are optimized and fixed one after the other according to their importance. In the problem under consideration, for example, the minimum total tardiness could firstly be determined and then the corresponding optimum energy costs are calculated. Conversely, it could also be possible to minimize TT for the lowest TEC calculated at the beginning. Both possible lexicographic solutions are exemplarily shown in Fig. 3.3 with black dots.

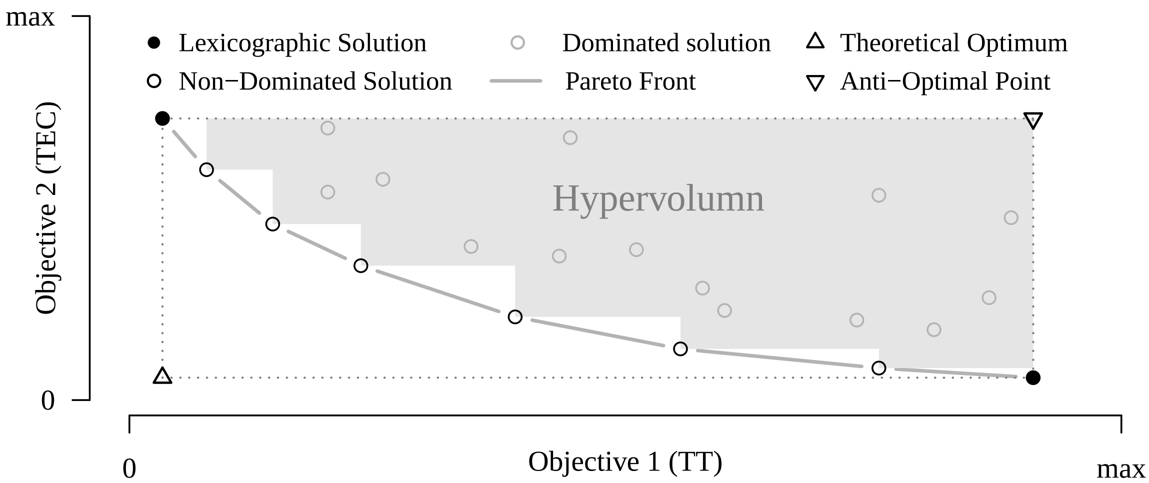


Figure 3.3: Visualization of lexicographic solution and pareto front with two objective functions

In general, all pareto optimal solutions come into question for the decision-maker. These solutions are also referred to as non-dominated solutions (NDS). A solution is dominated if another solution is just as good in all objectives and at least in one better. Vice versa, a solution is an NDS if it is better in at least one criterion compared to any other solution. The eps-constraint method is a suitable approach for calculating all pareto optimal solutions (pareto front) or at least some NDSs by means of mathematical modelling. Thereby, all lexicographical solutions are first identified. The range between these solutions is then analysed depending on the type of eps-constraint method. What all approaches have in common is that only one objective is optimized and all other objectives are limited by constraints.

In this article the equidistant eps-constraint method is pursued. Therefore, we calculate both lexicographic solutions. Then, TT is defined as a constraint at fixed intervals and the model is optimized for TEC as a single objective. This approach has two main weaknesses.

On the one hand, not every solution found is pareto optimal. The problem could be circumvented by the augmented eps-constraint method[59], in which the second objective is also included in the single objective function with a very small part. However, since almost every increase in TT leads to a reduction in energy costs and thus to an NDS, there is hardly any added value from the additional effort.

On the other hand, probably unevenly distributed pareto front is achieved. Here a dynamic approach like the bi-section eps-constraint method[60] could help, which always searches in the area where the euclidean distance between two NDSs is greatest. Since we limit the computing time of the models in the following, an optimal solution is not always found and the approaches can lead to different results. This, in turn, would lead to searches in different areas, which would make the evaluation of computing performance extremely difficult. Therefore, we apply the equidistant eps-constraint methode in the following.

3.4 Numerical case study

In the following, a numerical example shall illustrate the scheduling problem under consideration. Subsequently, a computational study is examined to evaluate the performance of the two proposed models. All problems are solved by CPLEX 12.6 using an Intel Xeon 3.3 GHz CPU with 768 GB memory. Even simple forms of HFS are considered to be NP-hard.[61] Consequently, only small instances can be solved for the described problem. The biggest problem instances that will be considered here consist of 10 jobs and 4 manufacturing stages. Real production processes often have much larger sizes. Nevertheless, the following example is constructed as realistic as possible.

[59]See e.g. Mavrotas (2009): *Effective implementation of the ε-constraint method*; Wang et al. (2018): *Bi-objective identical parallel machine scheduling.*
[60]Cf. Chircop / Zammit-Mangion (2013): *Epsilon-Constraint Based Methods.*
[61]Cf. Dai et al. (2013): *Energy-efficient scheduling.*

3.4.1 Test data

Initially, we consider six jobs which have to be processed at two stages whereby each stage consists of two parallel machines. The values for energy consumption and processing times are generated randomly from a uniform distribution U as follows:

- Processing Time $[h]$: $U\{1;10\}$,
- Energy Demand $[10^5 W]$: $U\{1;10\}$.

The exact values for the following example can be seen in Table 3.3.

Job	**1**	**2**	**3**	**4**	**5**	**6**
Processing Time $[h]$						
Stage1	3	8	7	3	9	7
Stage2	3	6	10	8	3	4
Energy Consumption $[10^5 W]$						
Stage1	6	1	3	10	8	7
Stage2	1	4	2	1	5	4
Due Date						
	19	17	12	10	6	6

Table 3.3: Numerical example

The processing speed of each job can be reduced up to 5 times which results in $l \in \{0,1,2,3,4,5\}$. Furthermore, all jobs have a due date D_j, which is also shown in Table 3.3. The definition of the due dates should be made in such a way that not all orders are automatically delayed, but it should also not be simply possible to solve the problem without a tardiness, as otherwise the scope for decision-making is restricted. Thus, their calculation is an essential factor influencing the complexity of the problem. On the basis of the work of Choi / Kim / Lee (2005)[62] we use the following formula:

$$D_j = \max\left(0, U\left[\left\lfloor P\left(1 - T - \frac{R}{2}\right)\right\rceil, \left\lfloor P\left(1 - T + \frac{R}{2}\right)\right\rceil\right]\right), \tag{3.31}$$

where P denotes makespan lower bound, T is a tardiness factor which influences the general amount of delays and R as due date range defines the scatter of D_j. Symbol $\lfloor ... \rceil$ indicates the nearest integer. For the first example we set T to 0.4 which leads to a fairly high average delay. R is set to 0.7 but will be varied in the following.

[62] Choi / Kim / Lee (2005): *Minimizing total tardiness.*

The estimation of the makespan is not only important to choose appropriate due dates, but also to reasonably limit the observation period. Too high values for the selected period under consideration of τ would lead to a situation where production would only take place in times of low energy costs at very low speeds, which leads to an enormous number of delays and does not represent a real alternative in practice. In detail τ is determined in equation (3.32). We calculate the average processing time at one stage k and add the maximum times of the previous and subsequent stages for one job. The optimal makespan definitely cannot exceed this value. Since the solutions for optimal TT does not necessarily lead to the optimal makespan and in addition, to allow enough margin for speed adjustments, the value is increased by a factor α. For the considered instances α is set to 0.1.

$$\tau = (1+\alpha) \cdot \min_{\forall k} \left[\max_{\forall j} \sum_{k^*=1}^{k-1} P_{jk^*} + \sum_{j=1}^{n} \frac{P_{jk}}{\mu_k} + \max_{\forall j} \sum_{k^*=k+1}^{m} P_{jk^*} \right] \tag{3.32}$$

Based on formula (3.32) a lower bound can be determined for the makespan. This corresponds to the average production time at a stage, extended by the minimum processing times at the other stages. Thus, P for equation (3.31) is calculated by:

$$P = \max_{\forall k} \left[\min_{\forall j} \sum_{k^*=1}^{k-1} P_{jk^*} + \sum_{j=1}^{n} \frac{P_{jk}}{\mu_k} + \min_{\forall j} \sum_{k^*=k+1}^{m} P_{jk^*} \right]. \tag{3.33}$$

Finally, TOU energy prices need to be defined. The average price per MWh for medium-sized industrial companies without major privileges in Germany is about 160 €[63]. In the following, this price will be set as mid-peak. For on- and off-peak deviations of 50% are assumed. Similar prices in USD are used for example by Ding et al. (2016)[64]. The exact prices depending on the corresponding time are shown in Table 3.4.

Hour	1 - 7	8 - 15	16 -20	21 - 22	23 - 24
TOU Price	80 €/MWh	160 €/MWh	240 €/MWh	160 €/MWh	80 €/MWh

Table 3.4: TOU prices for the numerical example

3.4.2 Evaluation of the example

To determine the optimal pareto front, we first have to define the lexicographic solutions. The corresponding schedules are shown in Fig. 3.4. For the considered problem minimum

[63] Cf. Fraunhofer (2015): *Electricity Costs of Energy Intensive Industries.*
[64] Ding et al. (2016): *Parallel machine scheduling under time-of-use electricity prices.*

TT is 36 h with energy costs of 4360 €. On the other side the minimum TEC can be found by 1351.73 € with TT of 103 h. Of course, both solutions differ significantly with regard to the two objectives but also in terms of computational effort. While the minimum TT solution can be found in less than 6 seconds by the time-indexed model, it takes 27.2 minutes to minimize TEC and to define the corresponding minimum TT. The more TT increases the higher is the number of possible speed reductions and time shifts for TOU prices. For minimum TT all processes are first performed at maximum speed which also reduces the CPU time.

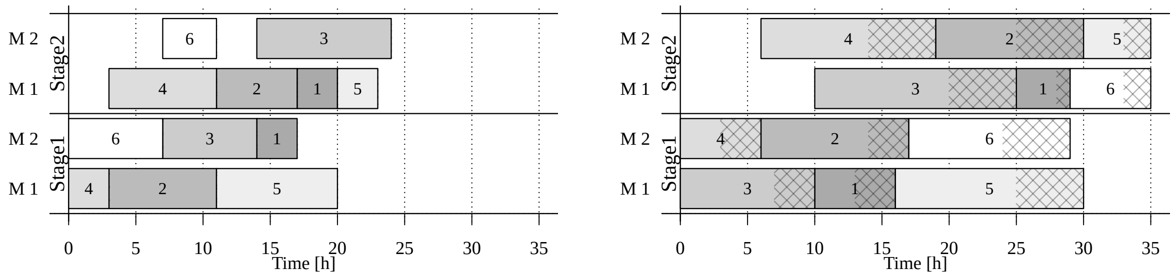


Figure 3.4: Optimal lexicographic solution for minimum TT (left) and minimum TEC (right)

Logically, also the energy consumptions distinguish significantly. Total Energy demand (TED) at minimum TT without any speed reductions lays at 28.4 MWh, while at minimum TEC only 10.1 MWh are needed. Similarly the time periods the energy is used differ which can be seen in Fig. 3.5. Not only the load curves but also the energy prices can be seen here. While for minimum TT TOU prices are barely used, for minimum TEC phases of higher energy consumption are mainly scheduled in times of lower energy prices (off-peak).

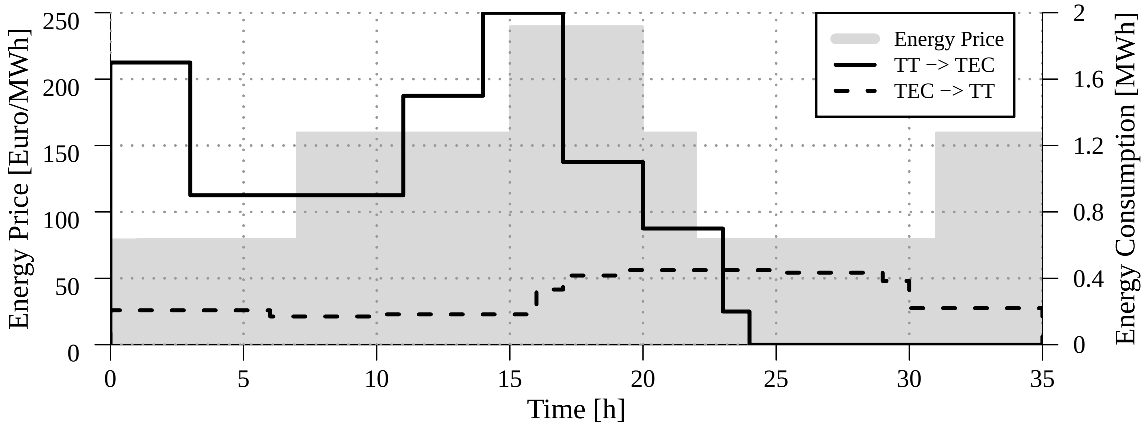


Figure 3.5: Energy price in Euro/MWh and load curves for lexicographic solutions

The two lexicographic solutions are only a small part of all pareto optimal solutions. By calculating the minimum TEC for all TT values between 36 and 103 we can determine the optimal pareto front and the corresponding energy demand shown in Fig. 3.6. Altogether 63 NDSs exist for the considered example with $\tau = 35$. The total energy costs can especially be reduced in the beginning when TT is very low. First additional delays allow to exploit the highest energy cost savings through speed reductions or making use of TOU prices. In the further process the curve flats gradually down.

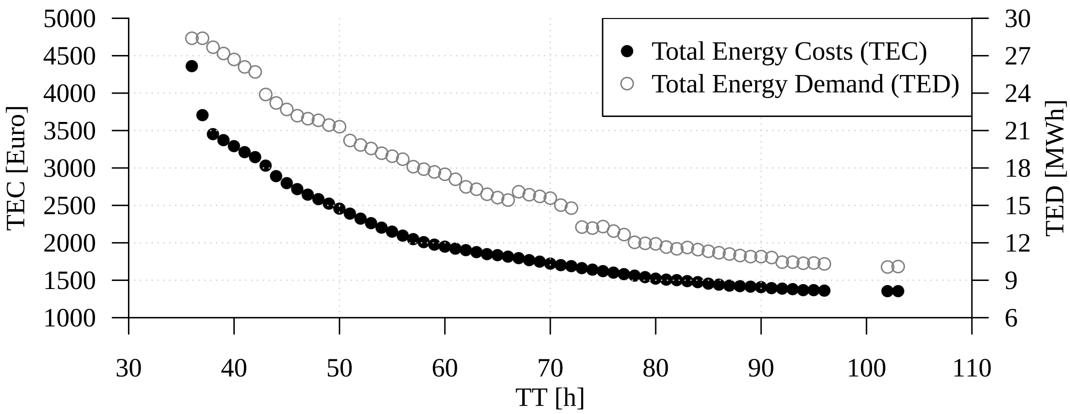


Figure 3.6: Optimal pareto front for the numerical example and resulting energy demand

Additionally, it can be seen that the course of TEC and TED are pretty similar. Nevertheless, a decreasing energy demand not necessarily leads to lower energy costs. Sometimes, energy demand even increases and the total costs can be reduced by shifting the consumption to times of lower TOU prices.

3.4.3 Performance analysis of different formulations

The computation of all pareto optimal solutions shown in Fig. 3.6 takes 8.48 h with the time-indexed formulation while the sequence-dependent formulation needs only 0.49 h. Thus, the calculation of all energy cost scenarios within a pre-process seems advantageous. In the following, differences in performance of both procedures shall be examined in detail and the relationship between the different objectives will be further discussed. Therefore, we generate test instances using the functions presented in section 3.4.1. The problem sizes are varied in the following according to Table 3.5. Altogether, 36 different problems are considered.

The performance of different models can be analysed in terms of model size complexity and computational complexity.[65] Regarding the model size Table 3.6 shows the number of

[65]See e.g. Meng et al. (2019): *MILP models for energy-aware flexible job shop.*

Parameter	Levels
Number of Jobs n	6, 8, 10
Number of Stages m	2, 4
Number of parallel Machines μ_k at each stage	2, 3
Due date Range R	0.4, 0.7, 1
Tardiness Factor T	0.4

Table 3.5: Overview of test instances

binary variables (NBV), number of continuous variables (NBC) and number of constraints (NC). The resulting model size for the test instances considered can be found in Table 3.7. The specified values for τ depend on the respective processing times and are calculated with equation (3.32).

Size	**Time-indexed formulation**	**Sequence-dependent formulation**
NBV	$nm(2\tau + o)$	$n\left(\sum_{k\in S}\mu_k + m(n-1+\tau(o+1))\right)$
NCV	$n(m(\tau+3)+1)$	$n(3m+1)$
NC	$nm(5+o+2\tau)+m\tau$	$n\left(m(n+5)+(n-1)\sum_{k\in S}\mu_k+1\right)$

Table 3.6: Model size in terms of number of variables and constraints.

Due to the variable e^t_{jk} in the time-indexed model, significantly more continuous variables are introduced. On the other hand, in the sequence-dependent formulation, the adjustment of the processing intensity is directly taken into account by z^t_{jkl}, which increases the number of binary variables enormously. With regard to the required constraints, considerably fewer equations are required for the second formulation.

How well the models are solvable cannot be inferred directly from the problem size but may be linked by analysing the computational complexity. Unfortunately, it takes a lot of computational effort to calculate all pareto optimal solutions. Therefore, computation time is limited to 10 minutes for each run. For that reason computation times will not differ as significantly as for the example above. However, the quality of the results should deviate. Furthermore, we calculate not all solutions but only a certain amount. Usually in eps-constraint method, the number of calculated solutions is set in the beginning. Thus, depending on the problem size the quality of the estimated pareto front can vary greatly. As already mentioned in section 3.3.5, we want to bypass that problem by using a predefined distance between two solutions. In detail we always increase TT by $\Delta = 5\,\mathrm{h}$. Thereby, the previous solution is always used as the initial solution for the next iteration. Thus, the pareto front can be appropriately estimated for each instance.

Since each problem instance has not only one optimal solution but different pareto optimal

Instance			Time-indexed			Sequence-dependent			τ
n	m	μ_k	NBV	NCV	NC	NBV	NCV	NC	
6	2	2	900	462	1030	2604	42	258	35
6	2	3	684	354	796	1968	42	318	26
6	4	2	2376	1206	2684	6936	78	510	47
6	4	3	2616	1326	2944	7680	78	630	52
8	2	2	1552	792	1724	4560	56	440	46
8	2	3	976	504	1112	2848	56	552	28
8	4	2	3936	1992	4332	11616	104	872	59
8	4	3	3360	1704	3720	9920	104	1096	50
10	2	2	1620	830	1796	4780	70	670	38
10	2	3	1460	750	1628	4320	70	850	34
10	4	2	4200	2130	4600	12440	130	1330	50
10	4	3	4520	2290	4936	13440	130	1690	54

Table 3.7: Model size of the considered instances

solutions, we first have to define a performance criterion to compare both approaches. In multi-objective optimization various concepts have been established. Probably the most frequently used criterion is the number of NDSs (see Table 3.8). Theoretically, the number of NDSs can be derived from the difference between the two TT values of the lexicographic solutions divided by 5 (since TT is increased by 5 h in each iteration). However, the solver does not always succeed in reducing energy costs within 10 minutes, which is why no new NDS is created in some cases. This results in different numbers of NDSs between the two approaches. These deviations are rather small and thus, this criterion is not very meaningful.

A much better criterion is the hypervolumn (HV)[66], which is visualized in Fig. 3.3. The idea is, to calculate the relative amount of space between theoretical optimum and anti-optimal point, which is covered by the found NDSs. Thereby, the theoretical optimum is the combination of the best objective values in the lexicographic solutions and vice versa the highest values for TT and TEC in the lexicographic solutions form the anti-optimal point. Therefore, HV has a value between 0 and 1, whereby higher values stand for a better solution quality since a larger area of the potential solution space is covered. In detail, a value above 0.5 means that the pareto front is convex between the lexicographic solutions. The higher the HV value, the more convexly curved is the pareto front and thus one gets closer to the theoretically optimal solution.

All HV values listed in Table 3.8 are additionally shown in Fig. 3.7. There is no instance

[66] See e.g. Beume / Naujoks / Emmerich (2007): *Dominated hypervolume.*

Ins-	Problem Size				Time-indexed			Sequence-dependent		
tance	n	m	μ_k	R	CPU [h]	NDS	HV	CPU [h]	NDS	HV
1	6	2	2	4	1.64	15	0.757	**0.156**	15	**0.758**
2	6	2	2	7	1.546	14	0.747	**0.126**	14	0.747
3	6	2	2	10	1.334	12	0.698	**0.117**	12	**0.699**
4	6	2	3	4	0.079	11	0.629	**0.007**	11	0.629
5	6	2	3	7	0.077	10	0.601	**0.007**	10	0.601
6	6	2	3	10	0.077	10	0.601	**0.006**	10	0.601
7	6	4	2	4	3.22	**19**	0.666	**1.344**	18	**0.709**
8	6	4	2	7	2.933	17	0.616	**1.192**	17	**0.677**
9	6	4	2	10	3.407	**20**	0.692	**1.318**	17	**0.728**
10	6	4	3	4	2.974	19	0.66	**0.146**	**20**	**0.677**
11	6	4	3	7	3.066	19	0.655	**0.156**	19	**0.674**
12	6	4	3	10	3.049	19	0.663	**0.144**	**20**	**0.677**
13	8	2	2	4	2.7	**15**	0.665	**2.678**	14	**0.695**
14	8	2	2	7	**3.027**	16	0.667	3.238	**17**	**0.707**
15	8	2	2	10	3.234	11	0.702	**2.963**	**16**	**0.753**
16	8	2	3	4	0.813	15	0.721	**0.146**	15	0.721
17	8	2	3	7	0.863	16	0.738	**0.131**	16	0.738
18	8	2	3	10	0.657	14	0.728	**0.118**	14	0.728
19	8	4	2	4	**5.517**	29	0.695	5.851	**32**	**0.793**
20	8	4	2	7	**5.551**	30	0.679	5.74	**32**	**0.783**
21	8	4	2	10	**5.596**	32	0.626	5.64	32	**0.74**
22	8	4	3	4	4.852	**27**	0.664	**3.259**	26	**0.734**
23	8	4	3	7	4.994	17	0.678	**3.768**	**27**	**0.747**
24	8	4	3	10	4.772	**27**	0.696	**3.025**	26	**0.743**
25	10	2	2	4	**2.116**	11	0.648	2.561	**13**	**0.692**
26	10	2	2	7	2.678	**15**	0.7	**2.57**	14	**0.741**
27	10	2	2	10	**2.607**	15	0.628	2.694	15	**0.693**
28	10	2	3	4	3.695	20	0.714	**3.488**	**21**	**0.738**
29	10	2	3	7	3.414	**19**	0.722	**2.897**	16	**0.737**
30	10	2	3	10	3.451	21	0.758	**2.437**	**23**	**0.771**
31	10	4	2	4	**3.34**	15	0.553	4.683	**25**	**0.768**
32	10	4	2	7	**2.336**	7	0.409	4.183	**22**	**0.666**
33	10	4	2	10	**2.671**	12	0.412	3.855	**20**	**0.634**
34	10	4	3	4	7.88	**44**	0.669	**7.534**	42	**0.769**
35	10	4	3	7	**5.226**	28	0.653	5.35	**29**	**0.739**
36	10	4	3	10	6.046	**33**	0.675	**4.516**	24	**0.773**
Average Values					**3.096**	**18.7**	**0.661**	**2.446**	**19.8**	**0.716**

Table 3.8: Detailed results of the computational study

where the time-indexed model finds a better pareto front than the sequence-dependent formulation. With an average of 71.6 % the second approach covers on average around 5 % more of the solution space. Since 7 of the first 18 instances can be solved to optimality with the made adjustments, both approaches lead to the same results in theses cases. It is noticeable that this occurs mainly when two production stages with three parallel machines are considered. As is usual within HFS problems, the computational effort increases with the number of stages and decreases with the number of parallel machines. This can also be identified to a limited extent by the computing time in the results.

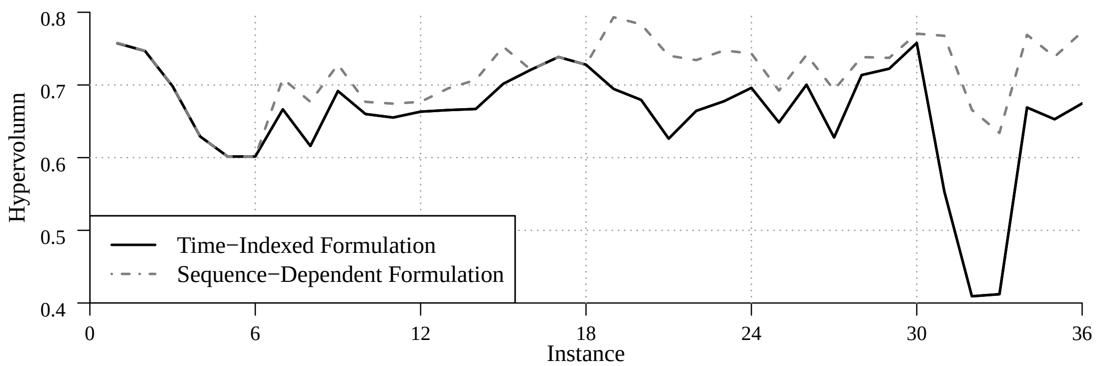
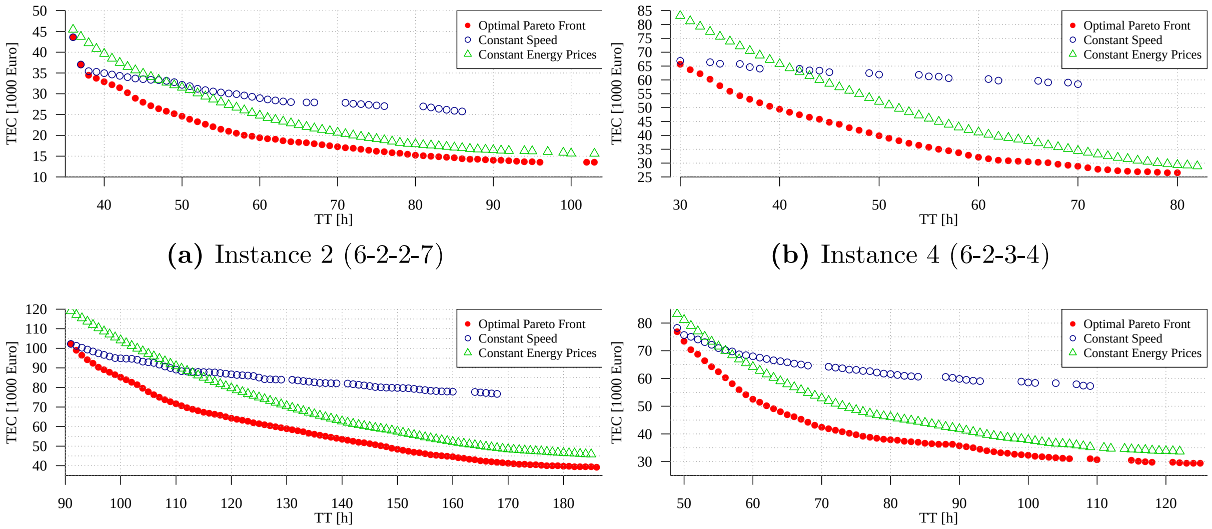


Figure 3.7: Hypervolumn results for the two models

A closer look at the computing times in Table 3.8 shows that the sequence-dependent model also has advantages here. However, the values must be viewed with caution. For larger problems, each optimization run is aborted after 10 minutes, which is why the times here differ less. It can also be seen that the CPU times are proportional to the number of NDSs. The time-indexed model leads especially to high computing times for the proof of optimality. As a result, the sequence-dependent method has significant advantages in computing time here. For example, for instances with 6 jobs, the average time of 0.39 hours is reduced by 80 % compared to the first model with 1.95 hours. Interestingly the due date range R has no general influence on the used performance indicators and computation time. Concerning the fluctuation of the results, the sequence-dependent procedure seems to work more robustly. For example, the standard deviation of the HV values of the time-indexed model is 7.66 % while the sequence-dependent procedure varies only by 4.88 %. Overall, the sequence-dependent approach appears much more suitable for the problem under consideration.

3.4.4 Evaluation of savings potentials

Finally, the possibilities of reducing energy costs will be explicitly analyzed. In particular, the influence of speed changes and TOU tariffs on the reduction of energy costs will be examined. Therefore, the model is solved once with constant (maximum) speed and once with fixed energy costs (160 €/MWh). The sequence-dependent model formulation is used exclusively for this purpose. Unfortunately, not all 36 pareto fronts can be shown here and only the first 18 instances can be solved to optimality in reasonable computing time. For illustration, we will concentrate in the following on instance 2 from section 3.4.2 and three further randomly selected instances (4, 12, 17). The results can be seen in Fig. 3.8.



(a) Instance 2 (6-2-2-7)

(b) Instance 4 (6-2-3-4)

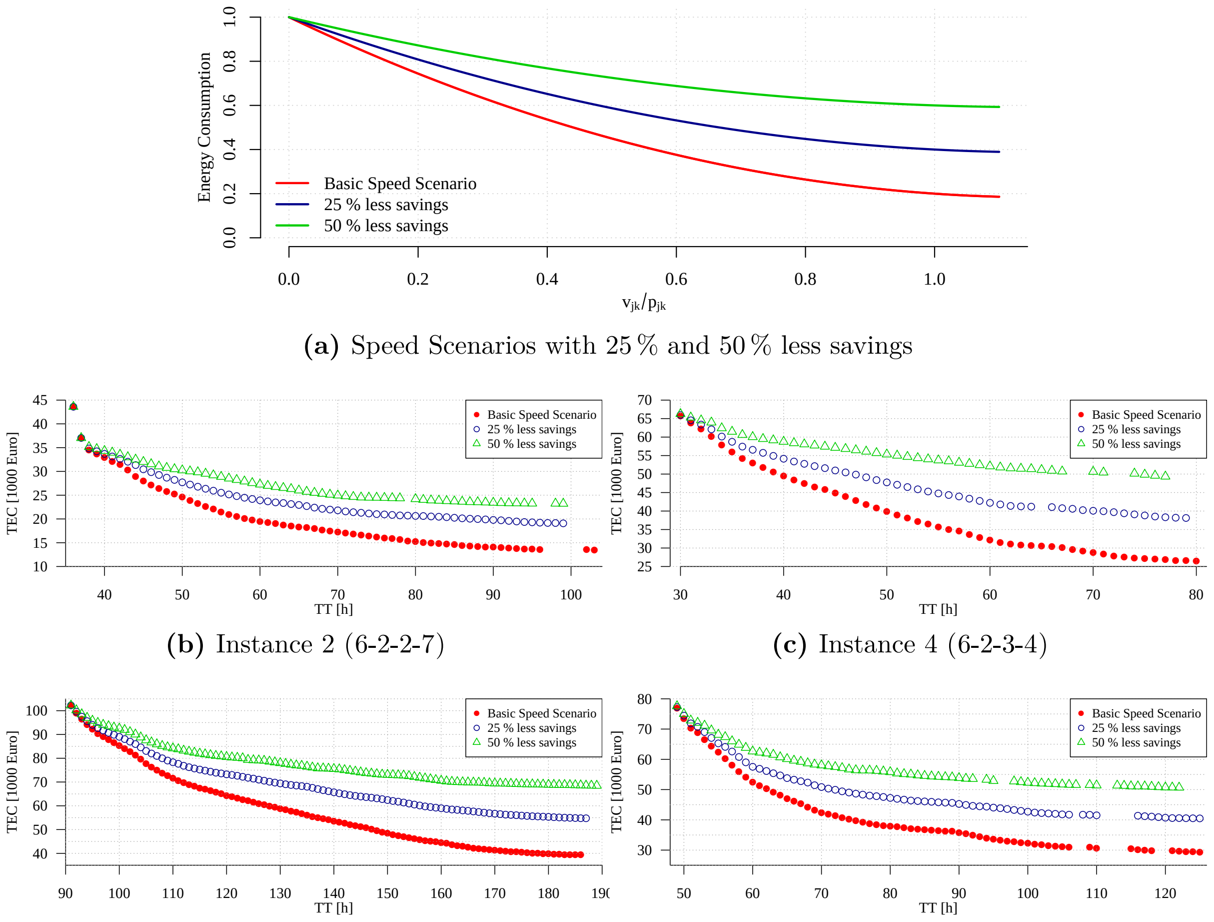
(c) Instance 12 (6-4-3-10)

(d) Instance 17 (8-2-3-7)

Figure 3.8: Pareto front for constant speed and constant energy prices

All points shown in Fig. 3.8 are pareto optimal solutions. The filled circles are the results when the original model is used. For the case of producing at maximum speed (unfilled circles), the cost savings are significantly lower. In addition, the number of NDSs is reduced. If fixed prices are used instead of the TOU tariffs (triangles), the TEC for mininimal TT are slightly increased. In comparison, the adjustment in production speed leads to significant savings. The relative reduction of costs is even higher for fixed electricity prices than for TOU tariffs. Overall, greater savings can be achieved by varying the speed. This suggests that it is preferred to reduce electricity consumption over purchase at more favourable conditions, which is also more sustainable.

Since speed changes seem to have a greater influence, it is interesting to examine the extend of energy savings when the load curve of the electric motor has a different course. For this purpose, we consider two less advantageous consumers. Fig. 9a) also shows two curves with 25 % and 50 % less savings potential, respectively, in addition to the original function. These differences can be identified almost to the same extent in the resulting pareto fronts in Figure 3.9. For the four instances considered, similar tendencies through speed control can be observed – an average of 22.9 % less energy reduction for the 25 %-case and 42.7 % on average for the 50 %-case. Considering the consistent developments, it can thus be concluded that speed adjustments have a significant influence on the energy costs.



(a) Speed Scenarios with 25 % and 50 % less savings

(b) Instance 2 (6-2-2-7)

(c) Instance 4 (6-2-3-4)

(d) Instance 12 (6-4-3-10)

(e) Instance 17 (8-2-3-7)

Figure 3.9: Influence of less energy savings due to speed reductions

Finally, it shall be discussed which of the various NDSs is a good solution from a decision-maker's point of view. Classical approaches of multi-criteria decision making such as minimizing the distance to the utopia point may not justify the importance of TT. As mentioned in section 3.4.2, greater energy cost savings can already be achieved by small increases in TT. The average relative reductions of TEC and the associated ranges are listed in Table 3.9. For this purpose, the delay for all 36 instances is increased by up to 10 h, whereby each individual increase (by 1 h) is taken into account. The computing time is limited to ten minutes.

If the decision maker is willing to increase TT by one hour, energy costs can already be reduced on average by 3.8 %. For the second hour, further savings of 3 % are possible (see row "stepwise"). This potential falls significantly with further delays. At a certain point, an additional delay is no longer economically justifiable. It can be observed that the possibilities of cost reduction fluctuate strongly. The average coefficient of variation is 41.03 %. Nevertheless, it can also be seen from minimum and maximum that the potential for cost reduction decreases with growing TT.

	Average TEC reduction in % for increasing minimum TT by									
	$1\,h$	$2\,h$	$3\,h$	$4\,h$	$5\,h$	$6\,h$	$7\,h$	$8\,h$	$9\,h$	$10\,h$
Minimum	0.8	2.8	4.1	5.1	6.8	7.6	8.4	9.3	10	10.6
Mean	3.8	6.8	9.1	11.3	13.3	15.4	17.2	18.9	20.4	21.9
Stepwise	3.8	3	2.4	2.2	2	2	1.8	1.7	1.5	1.5
Maximum	15	20.9	22.7	24.5	26.3	29.6	31.6	33.7	35.9	37.7

Table 3.9: Average savings potential for small TT increases for the 36 instances

3.5 Conclusion

The present article combines the ideas of energy efficiency and delivery reliability in production scheduling. Two multi-objective MIP formulations are given for the HFS scheduling problem considering variable production speeds to reduce energy consumption at the expense of longer processing times. Energy costs can be reduced not only by variable speeds but also by taking advantage of fluctuating TOU energy prices. To solve the problem eps-constraint method is used. As far as we know, this is the first time that energy costs and tardiness are considered as objectives in a hybrid flow shop problem with the described properties.

A numerical case study shows that energy costs can be enormously reduced by just a few delays in delivery. In this case, energy cost savings reached 3.8 % on average after postponing by one hour. It can be stated, that energy costs can be reduced by shifting loads to times of lower energy prices without reducing consumption. However, speed control seems to have a much stronger impact on energy costs than load shifts due to TOU prices. This suggests that it is preferred to reduce electricity consumption over purchase at more favourable conditions. At the same time, speed changes are more ecological as they also reduce energy consumption.

Based on the second (sequence-dependent) model formulation, it can be shown that by calculating all energy cost scenarios in a pre-process, computation time can be enormously reduced. Regardless of objectives or which model is used, very good or even optimal solutions are found quickly by solver for the considered problems. However, the proof of optimality takes a lot of time. Due to the high complexity of the problem, the specification of good lower bounds seems to be very difficult. Nevertheless, it might be possible to improve the solution finding by determining better lower bounds in a future work.

Since the problem is NP-hard, the models are only able to solve small problem instances. It is appealing for future work to develop heuristic solutions in order to solve larger problems in reasonable computing time. Nevertheless, the models can provide practical insights. On the one hand it can deliver reference solutions for the development of heuristics. Also, existing heuristic solutions may be upgraded with possible improvements by solvers. Furthermore, larger problems may be divided into subproblems. For example, we may prioritize bottleneck machines, where sequences are to be optimized with the model. Moreover, decomposition methods can be used. For example, batches can be formed which consist of similar jobs. Once the problem size is reduced, exact solution method can be applied. Next, each batch of jobs is considered individually and broken up for detailed planning. Thus, the models described are not only relevant from a research point of view, but can also be useful in practice to combine low energy costs and punctual delivery.

4 Energy Aware Scheduling in Flexible Flow Shops with Hybrid Particle Swarm Optimization

Abstract

This paper integrates energy awareness in the flexible flow shop scheduling system, where two objectives are minimized simultaneously: total tardiness and electric power costs. We also consider practical settings including variable processing speeds and time-of-use (TOU) electricity prices. A novel hybrid particle swarm optimization (HPSO) algorithm is developed which incorporates several distinguishing features: Particles are represented based on job operation and machine assignment, which are updated directly in the discrete domain. More importantly, we introduce a multi-objective tabu search procedure and a position based crossover operator to balance global exploration and local exploitation. Experiments are conducted to verify the performance of the proposed HPSO algorithm compared to the well-known approaches in the literature. Results show the significance of HPSO in terms of the number and quality of non-dominated solutions and computational efficiency.

Acknowledgement

Major Revision: J. Ding et al. (2020): Energy Aware Scheduling in Flexible Flow Shops with Hybrid Particle Swarm Optimization. In: *Computers & Operations Research*, Submitted Manuscript - Second Revision.

4.1 Introduction

Due to increasing automation in various industries, demand for self-regulating production scheduling is growing considerably in recent years. Meanwhile, networking of machines and equipments creates an accurate database for optimizations in industrial manufacturing. This is a curse and a blessing at the same time since scheduling can be planned much more precisely, while the scope for decision-making expands significantly, which requires more efficient algorithms.

The flexible flow shop problem (FFSP), also known as hybrid flow shop problem, is a branch of production scheduling. It commonly occurs in manufacturing environments in which a set of jobs have to be processed in a series of stages where each stage consists of several parallel machines.[1] Machines at each stage can be identical, related or unrelated at all. Jobs have to be processed by one of the machines at each stage, following the same order. FFSPs are widely used in industrial production, especially in areas such as electronics, food or textiles.[2]

The FFSP is strongly NP-hard even in the case with only two processing stages when one stage has two parallel machines and the other one machine.[3] Moreover, the two special cases of FFSP where there is a single machine at each stage, known as flow shop, and the case where there is a single stage with several machines, known as parallel machine, are also NP-hard.[4] As a result of the complexity, also in research community, efforts are devoted to this merger of flow shop and parallel machine problem.

Due to the synchronization of supply chains with just in sequence delivery requirements, timeliness is a crucial success factor. Typical objectives of FFSP are makespan, total completion time, machine utilization rate, etc. In addition to those efficiency oriented ones, further objectives become important in production planning. Also ecological aspects gain remarkable attention. In particular, energy consumption is often taken into account in recent years.[5]

With energy serving as one of the most important production resources, industry accounts for around 37 % of global energy demand and particularly in the chemical and metalworking industries, energy costs can account for more than half of gross production costs.[6] In the paper industry, for example, as the fourth largest industrial energy consumer,

[1] Cf. Ruiz / Vazquez-Rodriguez (2010): *The hybrid flow shop scheduling problem.*
[2] Cf. Huang / Yu / Yang (2013): *Multiprocessor Flow Shop Scheduling Problem.*
[3] Cf. Gupta / Hariri / Potts (1997): *Scheduling a two-stage hybrid flow shop.*
[4] Cf. Dessouky / Dessouky / Verma (1998): *Flowshop scheduling with identical jobs.*
[5] See e.g. Gahm et al. (2016): *Energy-efficient scheduling*; Biel / Glock (2016): *Energy-efficient production planning.*
[6] Cf. IEA (2018): *World Energy Balances: Overview.*

around 3 kWh are required per kg, which is roughly equivalent to the daily human power.[7] Consequently, the paper industry is often considered in energy-oriented scheduling.[8] Large amounts of energy are also required in the automotive sector - one of the key component in German industry, where approximately 30 MWh for the production of a middle class car are needed, especially for forming and casting.[9] By reducing energy consumption, in turn, cost as well as environmental pollution can be decreased.

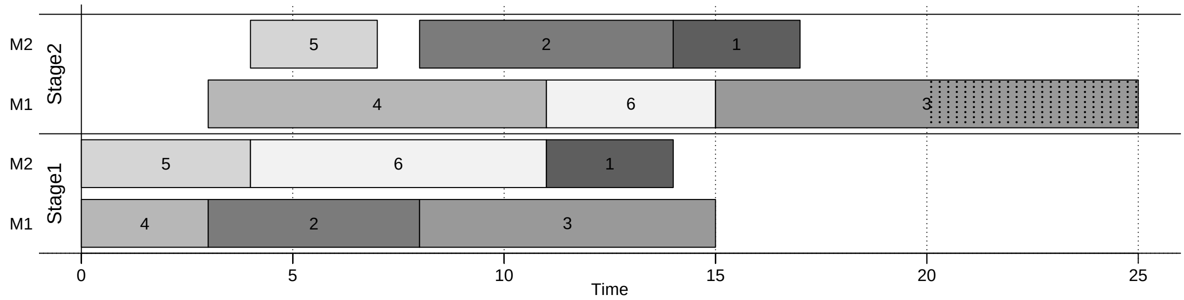
A sustainable energy supply is more effective in combination with flexible adjustment of demand in industry. Research is already being conducted in form of various joint projects. For example, WindNODE is a well known project with around 70 partners from industry and academia in Germany, which promotes load management in the industry to adapt to price fluctuations. The ultimate goal is to increase the amount of renewable energy by making consumers adapt to the supply. In particular, Siemens is investigating the possibility of shifting and adjusting loads in various plants, especially for energy-intensive processes, such as the balancing of turbines. By managing the timing and intensity, consumption and costs can be significantly reduced. At the same time, delivery commitments must be met.

In most practical cases concerning energy consumption, the problem automatically involves multiple objectives and measures. Usually, these objectives are mutually conflicting. Improving one objective often implies the deterioration of the other. In this study, we consider total tardiness (TT) and total energy costs (TEC) simultaneously.

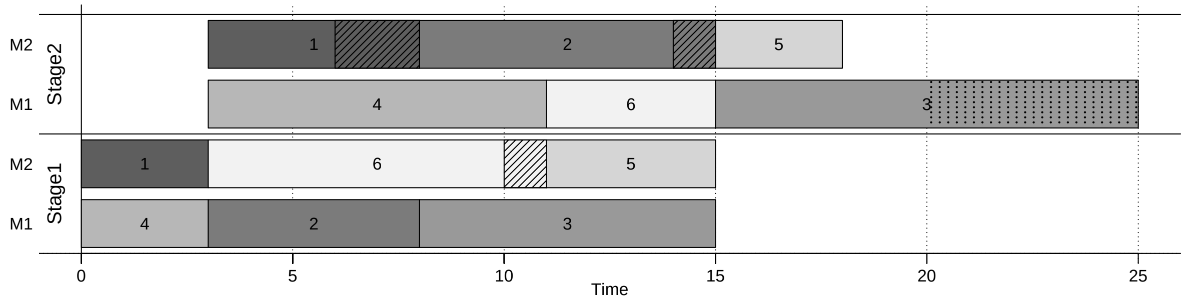
Energy costs are determined by multiplying the processing electricity consumption of the jobs with time-of-use (TOU) electricity prices. TOU electricity prices may vary from hour to hour depending on the time of the day. This offers the possibility of cost savings by shifting processes from hours of high prices (on-peak periods) to hours of lower prices (off-peak periods). Besides, jobs can be processed on each machines at different speed levels. When a machine is working at lower intensity/speed level, the energy consumption decreases but the processing time is prolonged. Therefore, under TOU scheme, TT and TEC are contradictory to each other. According to TOU, different starting times result in different energy costs. In order to reduce TEC, some operations may be delayed or prolonged to fit into periods of lower electricity prices. At the same time, this would increase TT.

Both possibilities of reducing energy costs are shown in Fig. 4.1. In this example, six jobs must be processed at two stages, each consisting of two parallel machines. Both schedules 4.1a and 4.1b lead to a minimum TT of 5 h. Delays are marked as dotted areas,

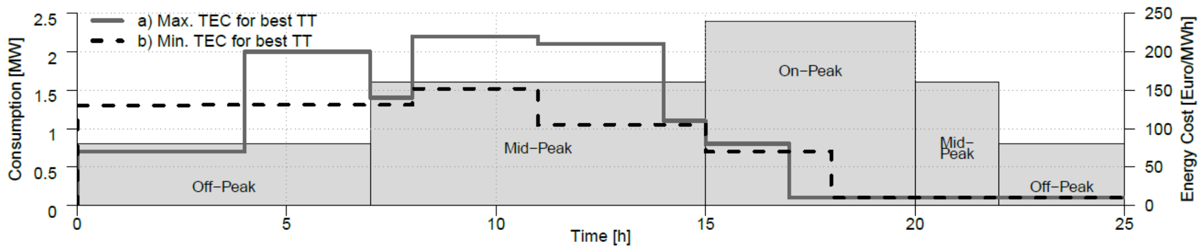
[7] Cf. Laurijssen / Faaij / Worrell (2013): *Benchmarking energy use in the paper industry.*
[8] See e.g. Zeng et al. (2018): *Multi-object optimization of flexible flow shop.*
[9] Cf. Smil (2016): *Embodied energy.*



(a) Highest *TEC* (3680 €) for minimum *TT* (5h)



(b) Lowest *TEC* (2949 €) for minimum *TT* (5h)



(c) Energy consumption for both solutions and TOU tariff (grey)

Figure 4.1: Numerical example to visualize the possibilities of *TEC* reduction

whereby the due dates of the 6 jobs are 17, 15, 20, 11, 18, and 15. However, by reducing the speed (hatched areas in Fig. 4.1b) the energy consumption can be reduced from 26.6 MWh to 22 MWh. Energy costs can then be further reduced by consuming more electricity at off-peak times (compare Fig. 4.1c). For that purpose, the more energy-intensive job 1 is brought forward in the schedule and job 5 is produced later. Since both jobs are not time-critical, this does not result in any additional delays.

The key issue of a solution is to assign each job to a machine, to select a processing speed level for the job, and to sequence the jobs on the machines for each stage. All these features bring great challenges to the manufacturing companies for reducing energy consumption while maintaining production efficiency. To the best of our knowledge, it is the first time that an efficient heuristic solution approach is proposed for this comprehensive problem. We develop a new hybrid particle swarm optimization (HPSO) algorithm, which allows high performance local search.

The rest of this paper is organized as follows. Section 4.2 summarizes literature on related scheduling problems. Section 4.3 describes the multi-objective flexible flow shop problem and introduces a mathematical formulation. Section 4.4 presents the main scheme and each component of HPSO. Computational results of HPSO and comparisons with the reference algorithms are reported in Section 4.5, while Section 4.6 concludes the paper and suggests future research directions.

4.2 Literature survey

There is a large amount of research work dealing with single objective FFSP. Extensive surveys are presented in Cheng / Guoqing Wang (1999)[10],Linn / Zhang (1999)[11],Kis / Pesch (2005)[12],Quadt / Kuhn (2007)[13],Ribas / Leisten / Framinan (2010)[14],Ruiz / Vazquez-Rodriguez (2010)[15],Lee / Loong (2019)[16], where exact and heuristic methods are investigated for the problem and its variants in the past decades. Especially in the area of FFSP scheduling, cases with more than one objective are rarely addressed.[17] Overall, less than a fifth of all FFSP articles are devoted to multi-criteria problems.[18] Among them, only a few articles present exact models due to their high complexity. For example, Sawik (2006)[19] of Sawik (2007)[20] solve the problem in several steps after dividing the time period by means of the master problem or by optimizing the machine assignment in advance.

The majority of publications in multi-objective FFSP discusses heuristic or metaheuristic approaches. Karimi / Zandieh / Karamooz (2010)[21] develop a multi-phase approach for bi-objective group scheduling in FFSP. Further multi-objective group scheduling articles are discussed in Neufeld / Gupta / Buscher (2015)[22]. A large number of multi-criteria FFSP contributions considers makespan as an essential objective. Marichelvam /

[10] Cheng / Guoqing Wang (1999): *A note on scheduling.*
[11] Linn / Zhang (1999): *Hybrid flow shop scheduling: A survey.*
[12] Kis / Pesch (2005): *A review of exact solution methods.*
[13] Quadt / Kuhn (2007): *A taxonomy of flexible flow line scheduling.*
[14] Ribas / Leisten / Framinan (2010): *Review and classification of hybrid flow shop scheduling.*
[15] Ruiz / Vazquez-Rodriguez (2010): *The hybrid flow shop scheduling problem.*
[16] Lee / Loong (2019): *A review of flexible flow shop.*
[17] Cf. Minella / Ruiz / Ciavotta (2008): *A review and evaluation of multiobjective algorithms.*
[18] Cf. Lee / Loong (2019): *A review of flexible flow shop.*
[19] Sawik (2006): *Hierarchical approach to production scheduling.*
[20] Sawik (2007): *A lexicographic approach to bi-objective scheduling.*
[21] Karimi / Zandieh / Karamooz (2010): *Bi-objective group scheduling.*
[22] Neufeld / Gupta / Buscher (2015): *A comprehensive review of group scheduling.*

Prabaharan / Yang (2013)[23] adapt a firefly algorithm to solve an FFSP with makespan and mean flow time as objectives. Also Ruiz / Allahverdi (2009)[24] examine a genetic algorithm for a bi-objective FFSP to minimize makespan and maximum tardiness. The same objectives are considered by Santosa / Riyanto (2016)[25] who present a hybrid of discrete differential evolution and variable neighbourhood search to solve an FFSP with resource-dependent processing times. Both total weighted tardiness (TWT) and the importance of customers are considered as objectives by Gonzalez-Neira et al. (2016)[26]. The authors use a greedy randomized adaptive search procedure in combination with simulation in a stochastic FFSP. Ebrahimi / Ghomi / Karimi (2014)[27] present two different genetic algorithms to minimize makespan and TWT under due date uncertainties. The same objectives are pursued by Li et al. (2015)[28], who also design a genetic algorithm in addition to a model.

Also in the area of energy aware FFSP some articles with multiple objectives can be found. Du et al. (2013)[29] propose an ant colony optimization metaheuristic considering both production efficiency and electric power cost with TOU prices. Tang et al. (2016)[30] apply an improved particle swarm optimization method in the dynamic FFSP to reduce energy consumption and makespan. Zeng et al. (2018)[31] develop a hybrid NSGA-II method to reduce makespan, electricity consumption, and material wastage. Zhang et al. (2014)[32] establish a time-indexed integer programming formulation for energy-conscious flow shop scheduling under TOU electricity tariffs. Dai et al. (2013)[33] propose an energy-efficient model and a genetic simulated annealing algorithm. Luo et al. (2018)[34] introduce a parallel genetic algorithm for solving an energy efficient dynamic FFSP using the peak power value with consideration of new arrival jobs. Makespan, total energy costs, and peak power are considered by Schulz / Neufeld / Buscher (2019)[35] who develop a multi-objective iterated local search algorithm with problem specific list scheduling algorithms.

Aiming to minimize TT and TEC simultaneously, already in Mouzon / Yildirim

[23] Marichelvam / Prabaharan / Yang (2013): *A discrete firefly algorithm.*
[24] Ruiz / Allahverdi (2009): *Minimizing the bicriteria of makespan and maximum tardiness.*
[25] Santosa / Riyanto (2016): *Hybrid differential evolution-variable neighborhood search.*
[26] Gonzalez-Neira et al. (2016): *Stochastic flexible flow shop scheduling.*
[27] Ebrahimi / Ghomi / Karimi (2014): *Hybrid flow shop scheduling.*
[28] Li et al. (2015): *A heuristic-search genetic algorithm.*
[29] Du et al. (2013): *Hybrid flow shop scheduling.*
[30] Tang et al. (2016): *Energy-efficient dynamic scheduling.*
[31] Zeng et al. (2018): *Multi-object optimization of flexible flow shop.*
[32] Zhang et al. (2014): *Energy-conscious flow shop scheduling.*
[33] Dai et al. (2013): *Energy-efficient scheduling.*
[34] Luo et al. (2018): *GPU based parallel genetic algorithm.*
[35] Schulz / Neufeld / Buscher (2019): *Comprehensive energy-aware hybrid flow shop.*

(2008)[36] looked at the two objectives TT and total energy demand (TED) in a single machine environment. Also for other scheduling problems like parallel machine scheduling,[37] batch scheduling[38] or job shop scheduling[39] different publications consider energy demand and tardiness. Time-dependent costs, on the other hand, are rather less researched.

In FFSP literature, only a few contributions optimize energy consumption and punctuality at the same time. Most multi-objective energy aware scheduling approaches consider makespan besides energy related objectives. Liu et al. (2014a)[40] develop an adaptive multi-objective genetic algorithm for a batch-processing machine scheduling problem and a hybrid flow shop problem to minimize TWT and energy related criteria. Jiang / Zhang (2019)[41] minimize non-processing energy besides TWT using an evolutionary algorithm combined with decomposition. In Nasiri et al. (2018)[42], NSGA-II and a non-dominated ranked genetic algorithm are compared to solve an FFSP with TWT and TED as objectives. The authors present an MIP formulation and consider sequence- and machine-dependent set-up times. A similar problem extended by fuzzy processing times is analysed by Zhou / Liu (2019)[43]. The authors propose an adapted multi-objective differential evolution algorithm to solve their introduced MIP formulation. To solve an FFSP with three objectives TT, TED and makespan, Li / Lei / Cai (2019)[44] suggest a two-level imperialist competitive algorithm.

All five mentioned contributions do not consider adaptable production speeds or time-dependent energy costs. Different discrete speeds are available in the FFSP problem settings of Lei / Gao / Zheng (2018)[45]. The authors discuss a teaching-learning-based algorithm to find pareto optimal solutions for TT and TED. Since TOU prices are not included, their search is limited to semi-active schedules. Time-dependent energy costs increase the complexity of scheduling problems enormously, as Chen / Zhang (2019)[46] show for the single machine case.

Overall, to the best of our knowledge, the only publication that examines the problem

[36] Mouzon / Yildirim (2008): *Minimise total energy consumption and total tardiness.*
[37] See e.g. Fang / Lin (2013): *Parallel-machine scheduling.*
[38] See e.g. Wang et al. (2016): *Batch scheduling.*
[39] See e.g. Liu et al. (2014a): *Minimising total energy consumption*; Zhang / Chiong (2016): *Solving the energy-efficient job shop.*
[40] Liu et al. (2014a): *Minimising total energy consumption.*
[41] Jiang / Zhang (2019): *Energy-oriented scheduling for hybrid flow shop.*
[42] Nasiri et al. (2018): *Minimizing the energy consumption and the total weighted tardiness for the flexible flowshop using NSGA-II and NRGA.*
[43] Zhou / Liu (2019): *Energy-efficient multi-objective scheduling.*
[44] Li / Lei / Cai (2019): *Two-level imperialist competitive algorithm for energy-efficient hybrid flow shop scheduling problem with relative importance of objectives.*
[45] Lei / Gao / Zheng (2018): *Teaching-learning-based optimization algorithm.*
[46] Chen / Zhang (2019): *Scheduling with time-of-use costs.*

considered here is Schulz / Buscher / Shen (2020)[47]. The authors analyse different MIP formulations and conclude that heuristic solution approaches are desirable. Our paper aims to address this research gap by presenting an efficient hybrid particle swarm optimization algorithm. The detailed problem properties are described in the following by introducing an MIP formulation which is based on Schulz / Buscher / Shen (2020).

4.3 Description of the considered FFSP

4.3.1 Problem setting

The multi-objective flexible flow shop problem with energy consumption can be described as follows: There are n jobs waiting to go through m stages following the same technological order. Each job $j \in \{1, \ldots, n\}$ has a due date d_j. If a job is completed later, the difference is called tardiness $T_j = \max\{0, C_{jm} - d_j\}$, whereby C_{jm} represents the completion time of job j at the last production stage m. The two objectives to be minimized simultaneously are total tardiness TT and total energy costs TEC. TOU tariffs are taken into account, which indicates that the price of electricity fluctuates over time.

Each stage $k \in \{1, \ldots, m\}$ consists of uniform parallel machines $l \in \{1, \ldots, m_k\}$ eligible for processing operations in the same stage. For each machine, there is a finite discrete set of processing speed levels. It is assumed that

- production can take place at different discrete processing speed levels;
- parallel machines are identical but can work at different speed levels at the same time, and
- the processing level is set for the duration of each task.

A reduction in production speed leads to lower energy consumption but prolongs the processing times. The speed selection can thus be represented by a parameter corresponding to additional processing time v_{jk}. For the operation of job j on stage k, denoted by o_{jk}, we define the baseline/minimum processing time p_{jk}. When o_{jk} is processed at a lower speed, which corresponds to a time increase of v_{jk}, the resulting processing time P_{jk} is determined by

$$P_{jk} = p_{jk} + v_{jk} \quad \forall j, k. \tag{4.1}$$

The stepwise increase of additional processing time follows the number of discrete production speeds available in the specific industry. Since a limited number of production speeds are physically possible, the processing time increases v_{jk} are selected from the set

[47] Schulz / Buscher / Shen (2020): *Multi-objective hybrid flow shop scheduling.*

$\mathcal{V} = \{0, \ldots, v_{max}\}$. It should also be pointed out that only integer values are taken into account. At the highest speed, v_{jk} is 0 since no additional machining time is introduced. The most energy efficient mode, however, leads to the maximum additional processing time $v_{jk} = v_{max}$. In other words, $0 \leq v_{jk} \leq v_{max}$ must apply.

According to the common $\alpha|\beta|\gamma$ coding[48], the flexible flow shop problem at hand can be summarized as

$$FF_m|P_{jk}(v_{jk}), d_j, TOU|TT, TEC.$$

Thereby, FF_m stands for a flexible flow shop with m production stages. The special features of the problem are the variable production times $P_{jk}(v_{jk})$, the consideration of due dates d_j, and the fluctuating TOU electricity prices. The third part refers to the two objective functions.

4.3.2 Mathematical model formulation

To formally describe the problem under consideration, the following notation is used:

Parameters

ep^t	= energy price per kWh in time period t
d_j	= due date of job j
e_{jk}	= baseline energy consumption of job j at stage k
es_{jk}	= energy saving factor depending on v_{jk} ($es_{jk} \in [0, 1)$)
p_{jk}	= baseline processing time of job j at stage k

Decision Variables

$$a^t_{jk} = \begin{cases} 1 & \text{if job } j \text{ is processed at stage } k \text{ in time } t \\ 0 & \text{else} \end{cases}$$

$$b^t_{jk} = \begin{cases} 1 & \text{if processing of job } j \text{ at stage } k \text{ starts in } t \\ 0 & \text{else} \end{cases}$$

C_{jk}	= completion time of job j at stage k
ec^t_{jk}	= energy consumption of job j at stage k in time t
v_{jk}	= proceesing time increase of job j at stage k

[48]For details see Graham et al. (1979): *Deterministic Sequencing and Scheduling.*

Based on the notation the problem under consideration can be formulated as an MIP as follows:

Minimize (I)

$$TT = \sum_{\forall j} \left(\max\left\{0, C_{jm} - d_j\right\}\right) \tag{4.2}$$

(II)

$$TEC = \sum_{\forall j}\sum_{\forall k}\sum_{\forall t} \left(ec_{jk}^t \cdot ep^t\right) \tag{4.3}$$

Subject to

$$\sum_{\forall j} a_{jk}^t \leq m_k \qquad \forall k, t \tag{4.4}$$

$$\sum_{\forall t} b_{jk}^t = 1 \qquad \forall j, k \tag{4.5}$$

$$b_{jk}^1 = a_{jk}^1 \qquad \forall j, k \tag{4.6}$$

$$b_{jk}^t \geq a_{jk}^t - a_{jk}^{t-1} \qquad \forall j, k, t > 1 \tag{4.7}$$

$$\sum_{\forall t} a_{jk}^t = p_{jk} + v_{jk} \qquad \forall j, k \tag{4.8}$$

$$ec_{jk}^t = \max\left\{0, e_{jk} \cdot \left(a_{jk}^t - es_{jk}(v_{jk})\right)\right\} \qquad \forall j, k, t \tag{4.9}$$

$$C_{jk} = \sum_{\forall t}(b_{jk}^t \cdot t) + p_{jk} + v_{jk} - 1 \qquad \forall j, k \tag{4.10}$$

$$C_{jk} \geq p_{jk} + v_{jk} + C_{jk-1} \qquad \forall j, k > 1 \tag{4.11}$$

The first two equations (4.2) and (4.3) are the objective functions to minimize TT and TEC. The solution space can be described by eight constraint sets. Constraint (4.4) ensures that each machine is assigned at most one job at a time. Since parallel machines are identical, no exact assignment to a machine needs to be made in the model. Jobs must not be interrupted, indicating that each job starts exactly once (constraint (4.5)). The connection between a_{jk}^t and b_{jk}^t is established by (4.6) and (4.7). Equation (4.8) ensures that an operation is scheduled for the entire processing time. This is composed of the baseline processing time and additional time due to speed reductions as already described in Eq. (4.1). If the speed for a process is reduced, the energy consumption decreases relatively by $es_{jk}(v_{jk}) \in [0, 1)$ depending on the additional processing time. Based on this assumption, the energy consumption can be calculated by (4.9). Constraint (4.10) defines the completion time C_{jk} of job j by its start time b_{jk}^t and real processing time. Since production already takes place at the starting time period, the term must be reduced by one time unit. Finally, (4.11) requires that a job is not processed at the next stage before the previous task has been completed.

Since the model formulation is time-indexed, the discrete time periods t must be restricted to the observation period $\{1, \ldots, \tau\}$. For the heuristic solution, on the other hand, it is not required to limit the observation period. For implementation and efficiency purposes, we have reformulated and introduced additional constraints. Especially referring to equation (4.9), an intermediate binary variable is defined. The problem can then be linearised and special ordered sets can be used. As the current formulation fully describes the problem, we do not go into details here.

4.4 Hybrid discrete particle swarm optimization algorithm

4.4.1 Discrete PSO

Particle Swarm Optimization (PSO) is a nature-inspired evolutionary computing algorithm motivated by the behavior of bird flocking and fish schooling. Originally proposed in Kennedy / Eberhart (1995)[49], it uses a stochastic search technique based on the simulation of social behavior metaphor, which exploits a population of potential solutions to probe the search space. PSO initializes the population with random candidate solutions, called particles. Each particle is assigned a randomized velocity and is iteratively moved through a multi-dimensional search space. During its flight, each particle improves its position according to its own experience and the experience of its neighbors. Combining with operators of evolutionary algorithms, PSO diversifies well and thus finds good solutions very efficiently.[50] Consequently, PSO approaches are already successfully employed for various multi-objective FFSP.[51]

Since the standard PSO is only suitable for the optimization problems in continuous space, a variant of PSO, Discrete Particle Swarm Optimization (DPSO) is proposed in Kennedy / Eberhart (1997)[52] for the combinatorial optimization problems in discrete space, in which the trajectories of particles are defined as the changes in the probability and the velocity is transformed from real number space to probability space via a sigmoid function. Given the inertia weight w and acceleration coefficients c_1 and c_2, the i-th particle at iteration t (i.e., X_i^t) can be updated as follows:

$$X_i^t = c_2 \otimes F_2(c_1 \otimes F_2(w \otimes F_1(X_i^{t-1}), P_i^{t-1}), G^{t-1}). \tag{4.12}$$

where P_i^{t-1} represents the local optimum of particle X_i during the past $t-1$ iterations,

[49] Kennedy / Eberhart (1995): *Particle swarm optimization.*
[50] Cf. Wang et al. (2010): *Improved hybrid discrete PSO.*
[51] See e.g. Ramezanian / Sanami / Nikabadi (2017): *Simultaneous planning of production and scheduling*; Tang et al. (2016): *Energy-efficient dynamic scheduling.*
[52] Kennedy / Eberhart (1997): *Particle swarm algorithm.*

and G^{t-1} is the global optimum of all the particles during the past $t-1$ iterations. This position updating equation consists of three components. The first component is the mutation of the particle itself, which is

$$\lambda_i^t = w \otimes F_1(X_i^{t-1}) = \begin{cases} F_1(X_i^{t-1}) & \text{if } rand() < w \\ X_i^{t-1} & \text{else,} \end{cases} \tag{4.13}$$

where λ_i^t is an intermediate particle, and F_1 represents a mutation operator. The second component is the "cognition" part of the particle representing the private thinking of the particle itself, which is

$$\delta_i^t = c_1 \otimes F_2(\lambda_i^t, P_i^{t-1}) = \begin{cases} F_2(\lambda_i^t, P_i^{t-1}) & \text{if } rand() < c_1 \\ \lambda_i^t & \text{else,} \end{cases} \tag{4.14}$$

where δ_i^t is an intermediate particle, and F_2 represents a crossover operator. The third component is the "social" part of the particle representing the collaboration among particles, which is

$$X_i^t = c_2 \otimes F_2(\delta_i^t, G^{t-1}) = \begin{cases} F_2(\delta_i^t, G^{t-1}) & \text{if } rand() < c_2 \\ \delta_i^t & \text{else.} \end{cases} \tag{4.15}$$

4.4.2 Hybrid discrete PSO algorithm

Similar to other nature-inspired evolutionary computing algorithms, the discrete PSO algorithm is capable of providing sufficient diversification during the search, while it often suffers from the drawback of being trapped into local optima.[53]

The main novelty of our algorithm lies in the effective local search algorithm employed into DPSO to intensify the search, which results in an (HPSO). The local search algorithm takes advantage of tabu search techniques. The general architecture of HPSO is presented in Algorithm 4.1.

As described in Algorithm 4.1, HPSO starts with a swarm of p particles generated by the *Init_Swarm* function. Initially, *pbest* is equal to each particle. HPSO then builds the archive (pareto) set A for the particles by the *Update_Archive* function, and randomly selects one solution in A as *gbest*. Then at each generation, HPSO

1) updates the position of each particle X_i^t according to Eq. (4.12),

2) applies *Tabu_Search* procedure to improve X_i^t, and

3) updates *pbest* and the archive set A with the newly optimized solution.

[53]Cf. Pan / Wang (2008): *No-idle permutation flow shop scheduling.*

Algorithm 4.1 Pseudo-code of the presented HPSO

Input: Instance of the considered problem
Output: The non-dominated solution set A found so far
$t \leftarrow 0$, $A \leftarrow \emptyset$
$\{X_1^t, ..., X_p^t\} \leftarrow$ Init_Swarm() /* Section 4.4.3 */
for $i = \{1, ..., p\}$ **do**
  $P_i^t \leftarrow X_i^t$ /* Init pbest */
  $A \leftarrow$ Update_Archive(X_i^t) /* Section 4.4.6 */
end for
/* Randomly select a solution from archive set A as gbest */
$k \leftarrow rand(0, 1, \ldots, |A| - 1), G^t \leftarrow A[k]$
repeat
  $t \leftarrow t + 1$
  for $i = \{1, ..., p\}$ **do**
    /*Update position according to Eq. (4.12)*/
    $X_i^t \leftarrow$ Update_Position($X_i^{t-1}, P_i^{t-1}, G^{t-1}$) /* Section 4.4.4 */
    $X_i^t \leftarrow$ Tabu_Search(X_i^t) /* Section 4.4.5 */
    $P_i^t \leftarrow$ Update_pbest(X_i^t) /* Section 4.4.6 */
    $A \leftarrow$ Update_Archive(X_i^t)
  end for
  $G^t \leftarrow$ Update_gbest(A) /* Section 4.4.6 */
until a termination condition is met
return A

After all particles are evolved, *gbest* is chosen from the current archive set. These procedures are repeated until a termination condition (usually a maximum run time T_{max} in seconds) is met. Note that *pbest* is the best previous local optimum of the particle, and *gbest* is the best global optima of all the particles.

4.4.3 Particle representation and swarm initialization

In order to perform the moves in the tabu search procedure, each solution is represented as a dual tuple $X = (S, V)$, where both S and V are three dimensional vectors ($n \cdot m_k \cdot m$). Each element $S[j][l][k]$ denotes a job including its position $[j]$ and assignment to machine $[l]$ at each stage $[k]$. Therefore, $S[j][l][k]$ stores simultaneously sequencing information as well as machine assignment. The additional processing duration is encoded by $V[j][l][k]$. Fig. 4.2 gives an example of the solution representation for an instance with 18 jobs and 2 stages, where each stage contains 2 machines. Fig. 4.2a shows the machine assignment and permutation of the jobs while Fig. 4.2b gives the speed selection of the jobs. For

	index	1	2	3	4	5	6	7	8	9	10	···	18
Stage 1	M_1	1	12	4	5	9	15	2	18	7		···	
	M_2	14	6	10	3	8	11	13	17	16		···	
Stage 2	M_1	3	9	11	5	17	18	1	6	13		···	
	M_2	4	8	7	12	2	15	10	16	14		···	

(a) Machine assignment and permutation of the jobs

	index	1	2	3	4	5	6	7	8	9	10	···	18
Stage 1	M_1	1	3	4	3	2	5	0	1	3		···	
	M_2	2	1	2	5	5	4	3	2	1		···	
Stage 2	M_1	2	4	0	1	0	2	1	2	3		···	
	M_2	4	1	3	4	2	1	3	1	1		···	

(b) Speed selection of the jobs

Figure 4.2: An example of solution representation in HPSO

example, at stage 1, job 11 is assigned to the 6-th position of M_2, and the corresponding processing level is 4. Therefore, we have $S[6][2][1] = 11$ and $V[6][2][1] = 4$ in $X = (S, V)$. Similarly for the same job on stage 2, $S[3][1][2] = 11$ and $V[3][1][2] = 0$.

The initial swarm is a population of candidate solutions which act as the starting point for the evolving of HPSO. In order to provide sufficient diversity for HPSO, the initial particles are generated randomly, i.e., each job is assigned to one machine at each stage with equal probability, and all jobs on the same machine are sequenced randomly. Besides, each job is assigned with an extra processing time randomly selected from the set $\mathcal{V} = \{0, \ldots, v_{max}\}$.

4.4.4 Position updating

The main purpose of position updating is to provide intensified transition to promising particles. This function can compensate the structural weakness of PSO as discussed earlier.

Conventionally for FFSP, a master sequence of all jobs is determined. According to this sequence, jobs are then assigned to machines by subordinate heuristic rules. Metaheuristic approaches usually operate on the master sequence. We intend to design a position updating strategy to break the master sequence restriction. The following neighborhood functions and genetic operators can achieve this purpose, which are integral parts of position updating.

Neighborhood definition

We first present several properties on potential movements, which help us to develop our neighbourhood structure. Given a solution X, the start time of each operation o_{jk} is denoted by s_{jk}. Therefore, linking to our MIP, $s_{jk} = \sum_{\forall t}(b^t_{jk} \cdot t)$ formally applies.

Definition 4.4.1 (Left side idle time) *Left side idle time I^l_{jk} of an operation o_{jk} is defined by*

$$I^l_{jk} = s_{jk} - \max\left\{C_{jk-1}, C_{[j-1]k}\right\} - 1. \tag{4.16}$$

By definition, for an operation o_{jk}, its left side idle time is determined according to the maximum completion time of its preceding operations of the same job C_{jk-1} and on the same machine $C_{[j-1]k}$. Similarly, we define right side idle time.

Definition 4.4.2 (Right side idle time) *Right side idle time I^r_{jk} of an operation o_{jk} is defined by*

$$I^r_{jk} = \min\left\{s_{jk+1}, s_{[j+1]k}\right\} - P_{jk} - s_{jk}. \tag{4.17}$$

Based on the newly introduced definition on idle time, it is possible to identify moves that provide immediate improvements.

Lemma 4.4.1 *Given a solution X, inserting an operation o_{jk} after $o_{j'k}$ leads to a dominating solution $X' \prec X$ if the following conditions are satisfied:*

1. $$C_{j'k} + 1 \leq s_{jk} + I^r_{jk} \tag{4.18}$$
2. $$P_{jk} \leq I^r_{j'k} \tag{4.19}$$
3. $$\sum_{t=C_{j'k}+1}^{C_{j'k}+P_{jk}} ep(t) - \sum_{t=s_{jk}}^{C_{jk}} ep(t) < 0. \tag{4.20}$$

Proof: With an insertion of o_{jk} directly after $o_{j'k}$, $s'_{jk} = C_{j'k} + 1$ holds. Condition (4.18) requires that operation o_{jk} as well as its successors are not postponed after the insertion. According to (4.19), none of the successors of $o_{j'k}$ is affected either. Therefore,

$TT(X') = TT(X)$ holds and energy costs remain unchanged for all operations except for o_{jk}. Condition (4.20) ensures that energy cost of o_{jk} is reduced which leads to $TEC(X') < TEC(X)$ and $X' \prec X$. Similarly, we can derive the following lemma for a swap movement.

Lemma 4.4.2 *Given a solution X, swapping operations o_{jk} and $o_{j'k}$ leads to a dominating solution $X' \prec X$ if the following conditions are satisfied:*

$$1. \quad s'_{jk} \leq s_{jk} + I^r_{jk} \tag{4.21}$$

$$2. \quad s'_{j'k} \geq s_{j'k} - I^l_{j'k} \tag{4.22}$$

$$3. \quad \sum_{t=s'_{jk}}^{C'_{jk}} ep(t) + \sum_{t=s'_{j'k}}^{C'_{j'k}} ep(t) - \sum_{t=s_{jk}}^{C_{jk}} ep(t) - \sum_{t=s_{j'k}}^{C_{j'k}} ep(t) < 0. \tag{4.23}$$

Depending on the assignment of o_{jk} and $o_{j'k}$, s'_{jk} and $s'_{j'k}$ can be further specified. The formulation given in Lemma 4.4.2 corresponds to the general case.

In addition, TEC can also benefit from switching processing levels.

Lemma 4.4.3 *Given a solution X, increasing additional processing time to $v'_{jk} > v_{jk}$ of an operation o_{jk} leads to a dominating solution $X' \prec X$ if the following conditions are satisfied:*

$$1. \quad v'_{jk} - v_{jk} \leq I^r_{jk} \tag{4.24}$$

$$2. \quad ec(v_{jk}) > ec(v'_{jk}) \left(1 + \frac{\sum_{t=C_{jk}+1}^{C_{jk}+v'_{jk}-v_{jk}} ep(t)}{\sum_{t=s_{jk}}^{C_{jk}} ep(t)}\right). \tag{4.25}$$

Proof. Following condition (4.24), no operations other than o_{jk} is delayed. For inequality (4.25), several transformation steps lead to

$$ec(v_{jk}) \cdot \sum_{t=s_{jk}}^{C_{jk}} ep(t) > ec(v'_{jk}) \sum_{t=s_{jk}}^{C_{jk}+v'_{jk}-v_{jk}} ep(t), \tag{4.26}$$

which ensures that the total energy cost of o_{jk} according to X' is reduced. Therefore, $X' \prec X$ is valid.

Lemmata 4.4.1–4.4.3 show that insertion, swap and speed changes can improve TEC without deteriorating TT. In accordance, we define three basic moves in the position updating and the local search procedure: *insert*, *swap*, and *speed*. In addition, we also relax conditions given in Lemmata 4.4.1–4.4.3, so that both TT and TEC can change simultaneously. This may not directly lead to dominating solutions, but it is necessary to reach new search spaces.

Given a solution X where operations o_{jk} and $o_{j'k}$ $(j \neq j')$ are jobs on machines l and l' of stage k with speeds v_{jk} and $v_{j'k}$, respectively, these moves can be defined as follows:

- $insert(X, o_{jk}, o_{j'k})$: Move o_{jk} out of l and insert it after $o_{j'k}$ of l';
- $swap(X, o_{jk}, o_{j'k})$: Swap o_{jk} and $o_{j'k}$;
- $speed(X, v_{jk}, v_{j'k})$: Change the processing speed of o_{jk} to $v_{j'k} \in \mathcal{V} \setminus \{v_{jk}\}$.

The neighboring solution sets $IN(X)$, $SW(X)$, and $SP(X)$ are generated by all the *insert*, *swap*, and *speed* moves performed on X, accordingly.

Note that the two operations in *insert* and *swap* moves are not confined to specific machines. If $l = l'$ holds, the moves focus on changing operation sequences. In the case of $l \neq l'$, machine re-assignments are involved. For a problem instance with n jobs, m stages and a candidate speed set of size $|\mathcal{V}|$, the maximum numbers of neighbors according to $IN(X)$, $SW(X)$, and $SP(X)$ structures are $O(n^2m)$, $O(n^2m)$, and $O(nm|\mathcal{V}|)$, respectively. As a result of *insert* and *swap* moves, no master sequence is present, and the resulting schedule becomes more flexible.

Mutation operator F_1

Mutation operator is used in the HPSO to perturb a solution to jump out of its local optima. In this paper, we adopt a simple mutation operator as the F_1 function in Eq. (4.12), i.e., applying $w \cdot n \cdot m$ times of *insert*, *swap*, and *speed* moves on the current solution, respectively, where w is a real number in the region $[0, 1]$. Note that all the operations in the moves are randomly selected.

Crossover operator F_2

The crossover operator in the HPSO, which acts as the F_2 function in Eq. (4.12), is used to assist the particles to learn from the previous experiences and experiences of other particles around them. Therefore, the crossover operator should be designed to inherit the good features and properties of the *pbest* and *gbest* particles. In this paper, we use a position based crossover operator (POX) which alternatively inherits the common jobs of the same machine from the parents to the offspring solutions and inserts the remaining operations to the vacant positions randomly. In detail, for a single production stage of the parent solutions, POX is performed as follows:

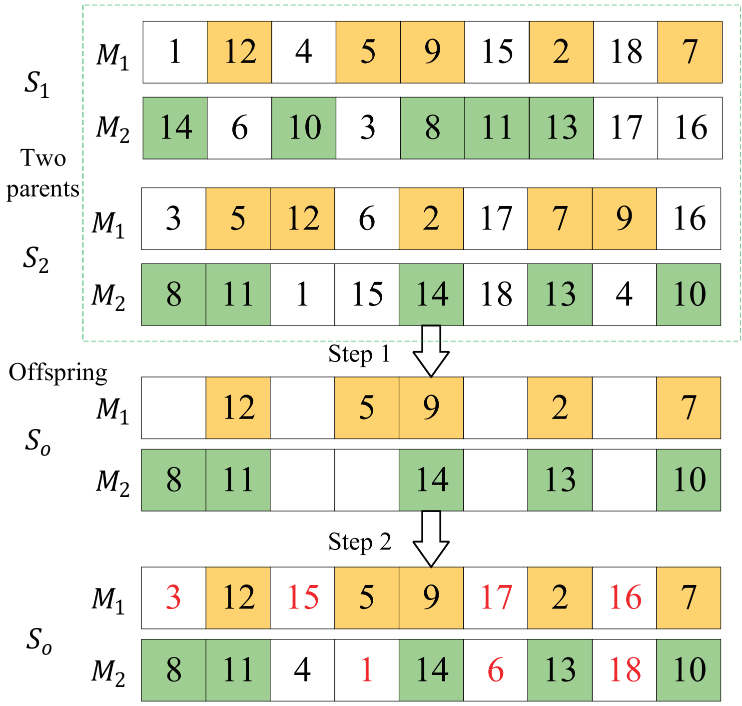


Figure 4.3: Illustration of crossover operator

(1) Identify the common jobs on the same machine of the two parent solutions, and alternatively inherit the common jobs from the machines of the parent solutions to the same machines of the offspring solution.

(2) For the remaining jobs, randomly insert them into the vacant positions of the offspring solution.

(3) If there are still jobs waiting to be assigned and all the vacant positions have been occupied, randomly append the jobs to the end of the machines.

(4) For common jobs, processing speed is inherited from one of the parents. For the remaining jobs, processing speed is chosen randomly.

By preserving common jobs while adding flexibility to different jobs, POX combines both greedy and random strategies to inherit the features from parents. Fig. 4.3 shows an example for the POX operator on the two parent solutions $X_1 = (S_1, V_1)$ and $X_2 = (S_2, V_2)$. S_1 and S_2 are job vectors with 18 jobs and 2 machines. It can be seen that $\{12, 5, 9, 2, 7\}$ as well as $\{14, 10, 8, 11, 13\}$ are the common jobs on M_1 and M_2 of S_1 and S_2, respectively. POX randomly starts with M_1 of S_1 and inherits the common jobs to the same positions on M_1 of offspring solution S_o, then alternatively inherits the common jobs on M_2 of S_2 to S_o. This procedure is repeated until all the common jobs are assigned to S_o. In the meantime, the corresponding processing speed of each common job in S_o remains consistent with its parents. Afterwards, the remaining jobs (marked in red) are randomly inserted into the vacant positions of S_o.

It should be noted that, when considering TOU prices, idle times may become desirable in the resulting schedule to reduce TEC. In previous research, dummy jobs are introduced to integrate idle times.[54] Extra idle time can also be inserted directly[55]. In contrast, vacant positions here can be used to imply idle times, when either the following job remains its starting time, or a vacant position takes fixed length. Afterwards, proper adjustments of speeds can occupy these positions and contribute to TEC. Therefore, this approach avoids handling additional jobs or idle times and can be very flexible.

4.4.5 Tabu search procedure

The tabu search procedure (TS) used in HPSO shall improve the solution quality by intensively exploring the search space. We apply the $f$irst improvement strategy in TS since multi-objective optimization aims to find a set of non-dominated solutions. For a given solution X, the proposed TS procedure sequentially operates on the $IN(X)$, $SW(X)$, and $SP(X)$ generated by *insert*, *swap*, and *speed* moves, respectively. In detail, TS iteratively improves the current solution by performing the first profitable move that is not in tabu status, then the corresponding move is recorded in the tabu list to prevent it from being selected during the next θ iterations (called tabu tenure). The tabu search procedure stops when it reaches the maximum number of iterations I_{max}, which is called the depth of the tabu search.

4.4.6 $pbest, gbest$, and archive updating

Solution *pbest* represents the best position of a particle along its search trajectories. Given a particle X_i and its *pbest* P_i, in order to facilitate implementation, the *pbest* is updated as follows: i) If X_i dominates P_i, P_i is replaced by X_i; ii) If P_i dominates X_i, P_i remains unchanged. iii) If X_i and P_i do not dominate each other, P_i is replaced by X_i with a probability of 0.5.

Particle *gbest* represents the best found position of all the particles so far. It is updated at the end of each generation by randomly selecting one solution from the archive set A. Thereby the archive set A preserves all the non-dominated solutions in the pareto front. Given a particle X_i, A is updated as follows: If A is empty, X_i is added to A. Otherwise, determine all the solutions in A which are dominated by X_i, and remove them from A, then add X_i to A. Consequently, each $X_i \in A$ represents a non-dominated solution.

[54]See e.g. Mansouri / Aktas (2016): *Minimizing energy consumption and makespan.*
[55]See e.g. Mansouri / Aktas / Besikci (2015): *Green scheduling of a two-machine flowshop.*

4.5 Computational results

4.5.1 Experimental protocol and benchmarks

In order to evaluate the performance of the proposed HPSO algorithm, we first compare the heuristic solutions with the MIP model presented in section 4.3 for small instances. The implementation is done with IBM ILOG CPLEX 12.7. Next, we compare HPSO with the well-known multi-objective evolutionary algorithm NSGA-II.[56] Both algorithms are implemented in C++ and run on an Intel Xeon E5-2697 processor with 2.60 GHz CPU and 2 GB RAM. Table 4.1 gives the descriptions and the settings for the main parameters of HPSO, which are decided by extensive preliminary experiments.

For parameter p, θ, I_{max}, and T_{max}, we use empirical values from previous studies. For the other parameters, there are generally two steps to determine their values: 1) Rough selection, which roughly selects proper values for the parameters from the domains of empirical values with large steps. 2) Parameter refinement, which refines each parameter from its domains with small step sizes. In both steps, we choose the value which gives the best performance in terms of both solution quality and computational efficiency.

Parameter	Section	Description	Value
p	4.4.2	Swarm size	10
w	4.4.2	Mutation rate	0.2
c_1	4.4.2	Recombination rate with *pbest*	0.7
c_2	4.4.2	Recombination rate with *gbest*	0.5
θ	4.4.5	Tabu tenure	$n + rand()\%n$
I_{max}	4.4.5	Depth of tabu search	10000
T_{max}	4.4.2	Maximum run time of HPSO	1200 seconds

Table 4.1: Parameter settings in HPSO

We take into account the following criteria for performance evaluation and comparison. These criteria include comprehensive measures adopted by common practice. Given are a non-dominated solution set A obtained by an algorithm and the reference front Re, which is the best known pareto front for the considered problem.

- CT: Converging time in seconds when no further improvement in solutions can be found.
- $NNDS$: The number of non-dominated solutions obtained by each algorithm.

[56] Cf. Deb et al. (2002): *NSGA-II.*

- $C(A, B)$: The coverage metric between two solution sets A and B, which denotes the percentage of solutions in B dominated by at least one solution in A, i.e.,

$$C(A, B) = |\{b \in B | \exists a \in A : a \prec b \text{ or } a = b\}|/|B|.$$

- $GD(A, Re)$: The generational distance which evaluates the average distance between the solutions in A and the reference front Re as follows:

$$GD(A, Re) = \frac{1}{|A|} \left(\sum_{a \in A} \min_{r \in Re} d(a, r)^p \right)^{1/p} \tag{4.27}$$

$$d(a, r) = \sqrt{\sum_{k=1}^{M} \left(\frac{a_k - r_k}{r_k^{max} - r_k^{min}} \right)^2}, \tag{4.28}$$

where $d(a, r)$ is the normalized Euclidean distance between two solutions a and r, a_k is the k-th objective of a, and p is an integer parameter which is set to 2 in this paper. Note that the reference front Re is the best pareto front obtained after conducting experiments on HPSO, HPSO-LS, NSGA-II, and CPLEX. Thus, Re corresponds to the set of NDS when all known solutions are considered together.
We normalize the distance because it helps to understand the resulting value. For example, 0.02 indicates that the NDS are on average 2% worse than the best pareto points found. Without normalization, the interpretation can be vague and confusing. A value of 5 could represent a good solution if the distance refers to TEC and vice versa very poor if the solutions differ by 5 units in TT.
- $S(A)$: The spacing metric is calculated with a relative distance measure between consecutive solutions in the obtained non-dominated set A, as follows:

$$S(A) = \sqrt{\frac{1}{|A|} \sum_{i=1}^{A} (d_i^{'} - \bar{d}^{'})^2} \tag{4.29}$$

$$d_i^{'} = \min_{k \in A \wedge k \neq i} \left(\sum_{m=1}^{M} |f_m^i - f_m^k| \right), \tag{4.30}$$

where $\bar{d}^{'}$ is the mean value of the above distance measure $\bar{d}^{'} = \sum_{i=1}^{|A|} d_i^{'}/|A|$. The distance measure is the minimum value of the sum of the absolute difference in objective function values between the i-th solution and any other solution in the obtained non-dominated set. Notice that this distance measure is different from the minimum Euclidean distance between two solutions. An algorithm having a smaller value of S is thus preferable.

4.5.2 Description of benchmarks

In order to evaluate the performance of the developed approach, we carry out extensive experiments. Furthermore, the interdependencies between tardiness, energy costs, as well as consumption and the choice of production speeds can be analysed. Unfortunately, to the best of our knowledge, there are no available test instances for the considered problem. For that reason, we generate random problem instances. The problem sizes are summarized in Table 4.2. We define two different problem categories. Small instances have up to 10 jobs, 4 stages, and 3 parallel machines. In this category, solver solutions can be found to verify the correctness of algorithms. For large instances we show the performance of the proposed heuristic for industrial sized problems.

Category	Problem class $n \cdot m \cdot m_k$			
	Jobs n	Stages m	Machines m_k	Replicates
Small	{6,8,10}	{2,4}	{2,3}	5
Large	{30,50,100}	{5,10}	{5,8}	5

Table 4.2: Descriptions of the benchmark sets

The baseline processing time p_{jk} and energy demand e_{jk} are randomly generated as follows:

- Baseline processing time p_{jk} [h]: $\sim U[1, 10]$
- Energy demand e_{jk} [$10^5 W$]: $\sim U[1, 10]$

Time Level	1-7h off-peak	8-15h mid-peak	16-20h on-peak	21-22h mid-peak	23-24h off-peak
Price	$80\frac{€}{\mathrm{MWh}}$	$160\frac{€}{\mathrm{MWh}}$	$240\frac{€}{\mathrm{MWh}}$	$160\frac{€}{\mathrm{MWh}}$	$80\frac{€}{\mathrm{MWh}}$

Table 4.3: TOU price levels

The data for TOU tariffs is shown in Table 4.3 and represents a typical winter day. The due date for each job j is determined by the following formula:

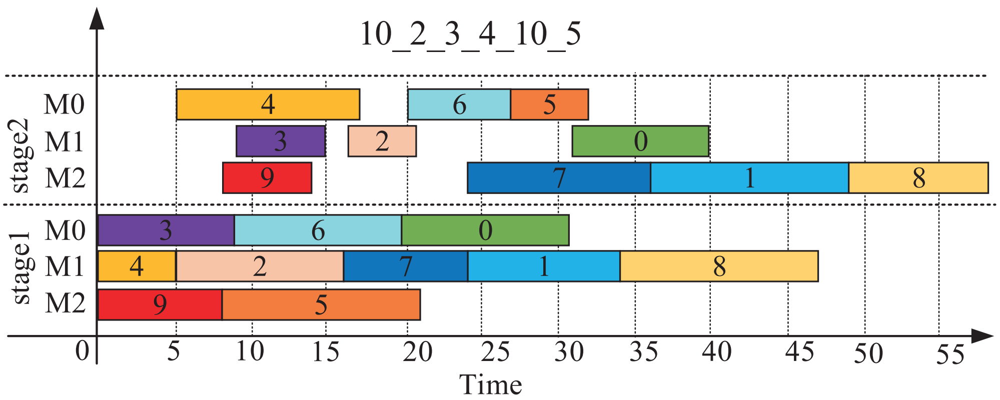
$$d_j = \max\left(0, U\left[\left\lfloor P\left(1 - T - \frac{R}{2}\right)\right\rceil, \left\lfloor P\left(1 - T + \frac{R}{2}\right)\right\rceil\right]\right), \tag{4.31}$$

where P denotes makespan lower bound, T tardiness factor, and R due date range, respectively. Symbol $\lfloor\rceil$ indicates the nearest integer. We set T to 0.4 which leads to a fairly high average delay. The due date range is varied with the values 0.4, 0.7, and 1.0.

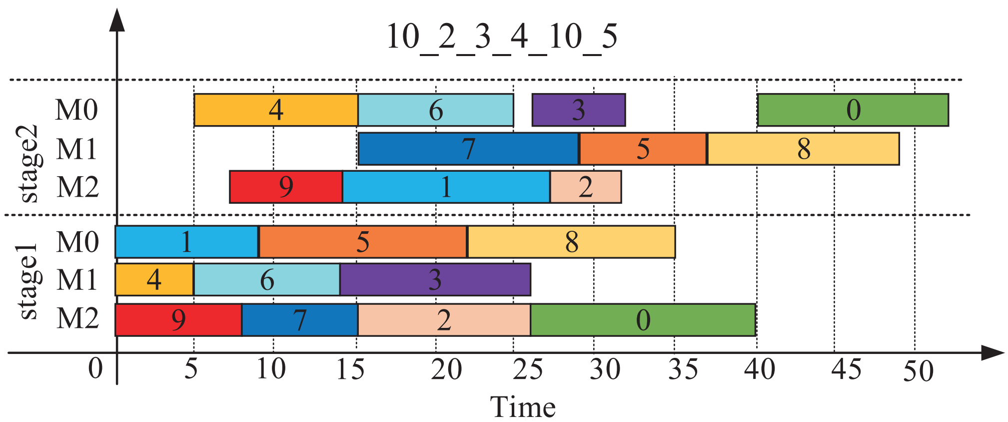
For each problem size we generate 5 random sample instances. Therefore, a total number of 360 problem instances are generated ($\#n \cdot \#m \cdot \#mk \cdot \#R \cdot Replicates$):

- Small: $3 \cdot 2 \cdot 2 \cdot 3 \cdot 5 = 180$,
- Large: $3 \cdot 2 \cdot 2 \cdot 3 \cdot 5 = 180$.

The aim of HPSO is to improve both objectives – the tardiness and the total electricity costs simultaneously. To visualize, Fig. 4.4 shows an example of two solutions for instance 10_2_3_10_5, where 10 jobs must be processed at 2 stages, each consisting of 3 parallel machines. Due date range R equals 1.0 and the sample number is 5. It is obvious that the local optimal solution (Fig. 4.4b) dominates the initial solution (Fig. 4.4a) with significantly smaller values of TT and TEC.



(a) An initial solution with TEC of 76392.7 € and TT 142 h



(b) An local optimal solution with TEC of 57355.2 € and TT of 94 h

Figure 4.4: The gantt charts of different solutions for instance 10_2_3_10_5

4.5.3 Comparison between HPSO and MIP model

First of all, we use the results generated by the MIP-model presented in section 4.3 to verify if HPSO is correctly implemented. Since the Branch and Cut algorithm used by the solver considers only one objective, we apply the equidistant epsilon constraint method for solution. Initially, the two lexicographic solutions (1. minimum TT and corresponding best TEC, 2. minimum TEC and corresponding best TT) are determined. Subsequently, the solver is used to minimize TEC for a given TT value defined by an additional constraint. By increasing TT by one each time (in the range between the two lexicographic solutions), the optimal pareto front can be determined.

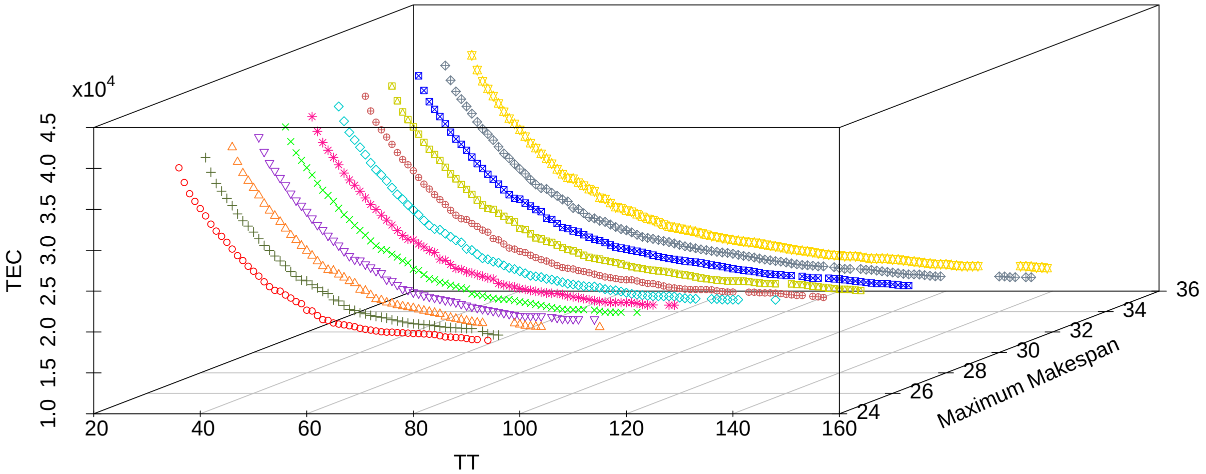
Since the model is time-indexed, the observation period τ must be limited. For that purpose, an upper bound for τ is calculated as shown in function (4.32). For each production stage, the total processing time is divided by the parallel machines and then the maximum time that a job can take before and after a given stage is added. For all calculated values, the minimum is taken into account and then increased by a percentage factor α.

$$\tau = (1+\alpha) \cdot \min_{\forall k} \left[\max_{\forall j} \sum_{k^*=1}^{k-1} p_{jk^*} + \sum_{j=1}^{n} \frac{p_{jk}}{m_k} + \max_{\forall j} \sum_{k^*=k+1}^{m} p_{jk^*} \right]. \tag{4.32}$$

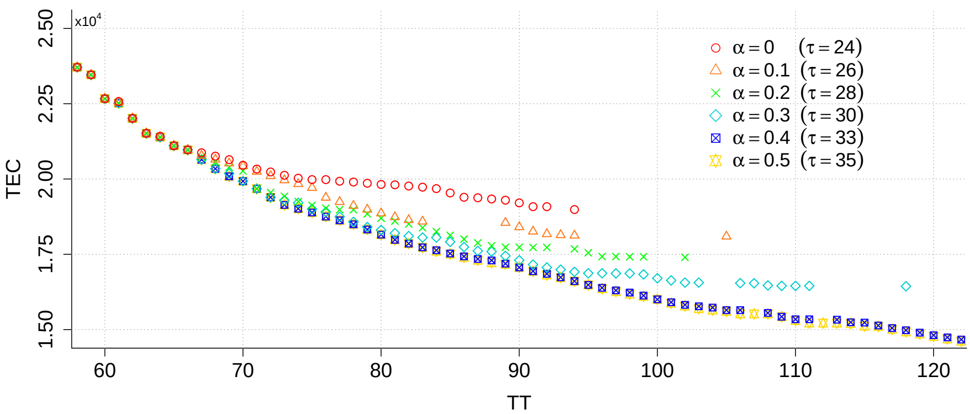
With smaller α, the number of variables can be reduced which makes the solution process easier for the solver. However, a small α also leads to the fact that periods with low TOU prices may be excluded and thus, some potential to reduce TEC is discarded. Note that, by definition, τ is equivalent to makespan. By varying the maximum τ in the model, the problem practically becomes three-dimensional with makespan as a third objective. As an example, Fig. 4.5 shows the optimal solutions with different α for instance 6_2_3_7_5 indicating 6 jobs, 2 stages, 3 parallel machines, due date range $R = 0.7$, and random sample number 5. The tri-objective case is given in Fig. 4.5a. When the maximum makespan (τ) is increased, more NDS can be found. On the other hand, the energy costs for higher TT restrictions can be significantly reduced. The latter can be seen clearly in Fig. 4.5b, which attributes the problem back to the bi-objective case and shows only the part where the $TECs$ differ ($TT > 60$). However, it must be noted that the results are improved at the expense of excessive computing time. The increase of the computing time depending on τ is shown in Table 4.4.

τ - max. Makespan	24	25	26	27	28	29	30	31	32	33	34	35
CPU time $[min]$	16.28	20.18	16.98	32.45	33.30	44.35	48.75	53.80	76.28	90.07	108.95	127.27

Table 4.4: CPU time for optimal pareto front of Instance 6_2_3_7_5 depending on τ



(a) 3D Problem with $\tau \in \{24, ..., 35\}$



(b) Zoom in 2D - Subset of NDS

Figure 4.5: Influence of α and τ on MIP solution for instance 6_2_3_7_5

For the first verification of the HPSO, Table 4.5 compares the lexicographic solutions of HPSO with CPLEX on the benchmark instances with 6, 8, and 10 jobs. Thereby, α is set to 10 % and the computation time for CPLEX is limited to 20 minutes for each run. The results are grouped by different values of n, m, and m_k, where each combination consists of 15 instances. Columns $\bowtie$, $\prec$, $\succ$, and $=$ respectively denote the number of instances where the extreme point of CPLEX does not dominate, dominates, is being dominated by, and is identical to the solutions obtained by HPSO.

Overall, the magnitudes of the solutions are very similar and, especially for small instances, both approaches usually find the same first lexicographic solutions. It is particularly noticeable that the heuristic results are better for the second lexicographic solution and the number of instances where HPSO dominates CPLEX increases with n, while the number of instances where HPSO is identical to the MIP model decreases. There are two primary reasons for this result:

n	m	m_k	CPLEX v.s. HPSO 1. lexicographic sol. (TT_{min}, TEC_{max})				CPLEX v.s. HPSO 2. lexicographic sol. (TT_{max}, TEC_{min})			
			$\bowtie$	$\prec$	$\succ$	$=$	$\bowtie$	$\prec$	$\succ$	$=$
6	2	2	0	0	0	15	0	3	12	0
		3	0	0	0	15	0	2	13	0
	4	2	2	7	0	6	0	0	15	0
		3	0	1	1	13	3	12	0	0
8	2	2	0	2	2	11	0	1	14	0
		3	0	1	1	13	2	1	12	0
	4	2	11	4	0	0	1	8	6	0
		3	10	4	0	1	1	13	1	0
10	2	2	2	3	7	3	0	0	15	0
		3	0	3	8	4	2	8	5	0
	4	2	1	0	14	0	1	1	13	0
		3	0	2	13	0	0	15	0	0

Table 4.5: The comparison results of HPSO and MIP model on the benchmark instances

1. The HPSO does not have to limit the observation period, and can thus find better solutions compared to the solver, as the latter operates within a time horizon, and does not consider possible lower TOU prices in later periods. This especially accounts for small α values.

2. The solver only proves optimal solutions for instances with 6 jobs. For larger problem instances, the optimization (especially of TEC) becomes exceedingly difficult within 20 minutes.

The differences can be alleviated by increasing the value of α. Fig. 4.6 shows all NDS found by both approaches for 4 small instances, when setting $\alpha = 0.5$. The four parts (a,b,c,d) in Fig. 4.6 are representative of the general results. It can be seen that major parts of the fronts of MIP model and HPSO overlap. With an increased α value (compared to Table 4.5), CPLEX has fewer restrictions and thus obtains more NDS. However, this leads to longer computing times due to larger solution space. In fact, if α is sufficiently large and computing time is long enough, solutions of CPLEX become dominant. Especially with greater TT values, HPSO deviates slightly from the CPLEX solutions (e.g., Fig. 4.6 (c)). But overall, HPSO can find a large portion of the optimal NDS. HPSO sometimes even dominates (e.g., Fig. 4.6 (d)), since CPLEX, despite the long running time, again encounters computing time limitations. In this respect, it is worth mentioning that HPSO generates the solutions in a fraction of a minute, while the solver requires several hours to complete the solution process for a single instance.

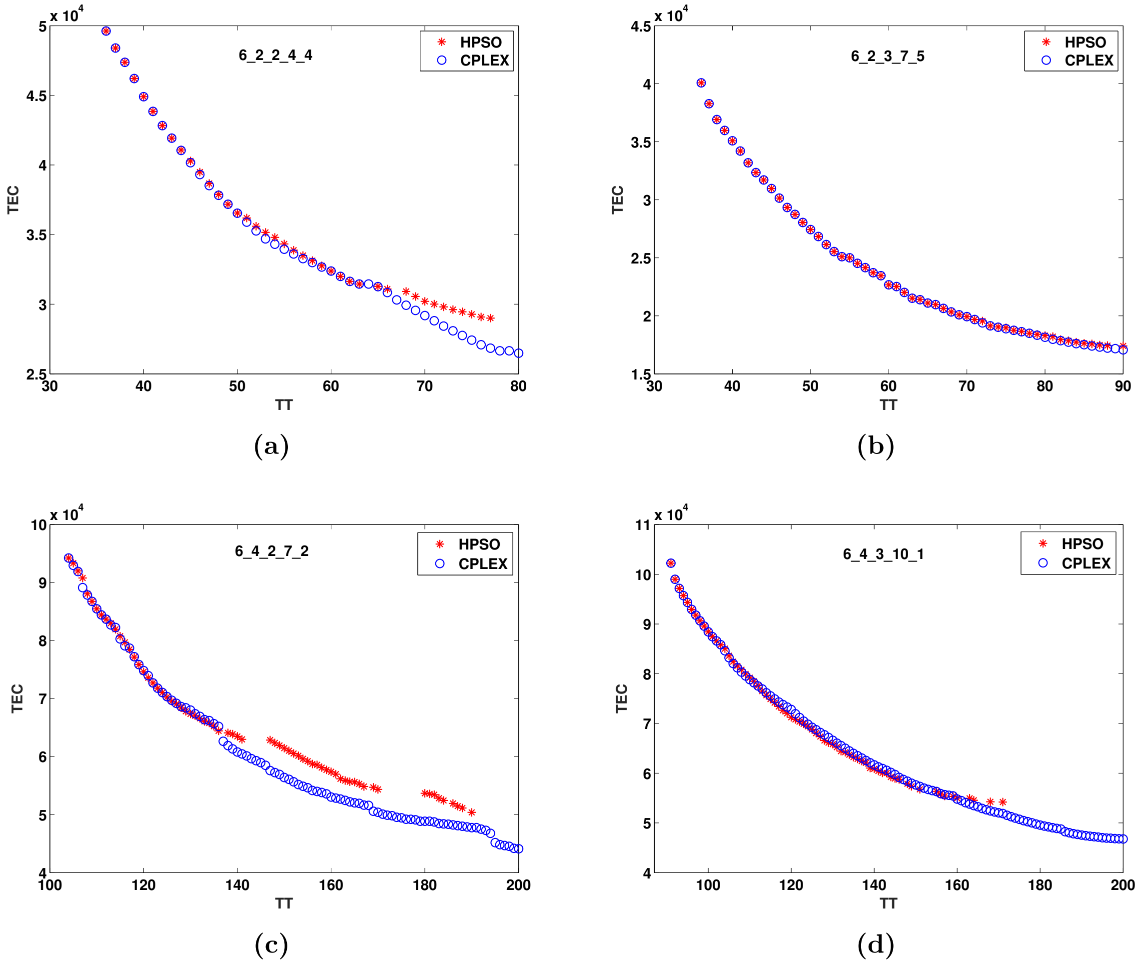


Figure 4.6: The distribution of the non-dominated solutions obtained by HPSO and CPLEX

4.5.4 The overall computational results

To verify the algorithm performance on a large scale, we compare HPSO with NSGA-II over 10 runs of all 360 test instances. Table 4.6 reports the average values of $NNDS$, S, C, GD, and CT grouped by different sizes of the benchmark instances. Columns n, m, m_k are the number of jobs, stages, and parallel machines. Column PF represents the best known pareto front which is obtained after running all test algorithms.

It can be observed from Table 4.6 that HPSO outperforms NSGA-II in terms of both solution quality and computational efficiency. The HPSO takes an average of 296.55 seconds to solve an instance while the NSGA-II exceeds that time by more than 50 %. Note that we use converging time CT to accurately measure the time necessary to reach the best known solutions. This value turns out to be much smaller than the maximum

n	m	m_k	PF		HPSO					NSGA-II				
			$NNDS$	S(PF)	$NNDS$	S(HPSO)	C(HPSO, NSGA-II)	GD(HPSO, PF)	CT	$NNDS$	S(NSGA-II)	C(NSGA-II, HPSO)	GD(NSGA-II, PF)	CT
6	2	2	62	65.72	62	65.54	1	0.02	15.32	49	72.96	0.94	0.07	30.54
		3	64	61.82	62	62.34	0.99	0.02	22.89	53	73.42	0.97	0.07	38.42
	4	2	82	65.9	74	71.93	1	0.03	37.6	58	88.05	0.75	0.06	53.58
		3	82	60.78	71	67.32	1	0.03	58.45	57	81.8	0.76	0.06	67.12
8	2	2	87	53.68	85	54.68	1	0.02	84.37	63	53.24	0.87	0.06	95.6
		3	68	68.63	64	71.5	1	0.02	88.54	58	67.92	0.93	0.07	103.58
	4	2	83	67.5	77	69.27	1	0.02	105.62	40	169.29	0.47	0.07	134.96
		3	92	68.29	75	79.54	1	0.02	95.4	46	110.12	0.44	0.05	159.77
10	2	2	87	49.43	80	53.27	1	0.02	114.2	65	52.79	0.8	0.07	187.25
		3	87	65.45	82	68.63	1	0.02	127.56	61	79.01	0.8	0.06	212.34
	4	2	79	101.65	72	109.46	1	0.02	152.36	28	295.71	0.25	0.08	274.65
		3	80	89.6	75	93.93	1	0.02	177.9	29	371.12	0.33	0.06	282.37
30	5	5	84	63.54	84	67.22	1	0.04	215.43	56	110.57	0.81	0.08	378.14
		8	75	57.12	75	59.36	1	0.03	258.67	48	98.66	0.75	0.08	415.38
	10	5	95	62.3	83	66.54	0.99	0.03	293.08	65	89.36	0.58	0.12	568.97
		8	108	55.64	101	63.15	0.97	0.04	327.56	83	93.47	0.64	0.15	585.33
50	5	5	98	48.2	98	49.55	1	0.02	338.7	69	78.12	0.83	0.11	649.52
		8	112	45.56	109	48.23	0.99	0.03	357.61	75	80.3	0.6	0.23	712.56
	10	5	85	61.25	80	65.7	0.98	0.03	438.75	52	125.41	0.52	0.24	878.31
		8	90	50.27	90	53.69	1	0.02	523.88	50	147.62	0.48	0.31	890.15
100	5	5	94	48.71	89	52.74	0.97	0.04	725.43	61	135.48	0.74	0.15	956.67
		8	103	56.42	103	58.97	1	0.02	774.59	70	187.36	0.62	0.15	982.15
	10	5	88	46.3	88	47.38	1	0.03	826.57	58	215.33	0.43	0.34	1038.28
		8	92	62.59	92	64.23	1	0.02	956.8	67	243.69	0.36	0.38	1178.51
Average			86.54	61.51	**82.13**	**65.17**	**0.99**	**0.03**	**296.55**	56.71	130.03	0.65	0.13	453.09

Table 4.6: The overall comparison results of HPSO and NSGA-II on the benchmark instances

computing time given T_{max}. Thereby, our algorithm determines an average of 82.13 NDS while NSGA-II finds 56.71. More importantly, the comparison of C(HPSO,NSGA-II) shows that at maximum 3 % and on average less than 1 % of the HPSO solutions are dominated by the NSGA-II. Conversely, over a third of the NSGA-II NDS are dominated by HPSO.

Furthermore, the average distance between the NDS found (spacing metric S) is remarkably smaller for the HPSO (65.17 compared to 130.03 for NSGA-II). The HPSO also performs considerably better compared to the best pareto front PF found. The last criterion GD(HPSO,PF) shows that the solutions of our approach deviate on average 3 % from PF while NSGA-II at 13 %. Moreover, the distance GD(NSGA-II,PF) increases with problem sizes, indicating that the performance of NSGA-II drops. Comparatively, the GD(HPSO,PF)-values remain constant between 0.02 and 0.04 independent of problem sizes.

In order to further analyze the performances of HPSO and NSGA-II, we conduct two sample t-tests on the two algorithms with respect to different criteria on all the 360 instances. The statistical results on GD and S are reported in Tables 4.7 and 4.8, which are grouped by different problem sizes. Columns $Mean$ and SD represent the mean value and standard deviation of GD and S, respectively. From Tables 4.7 and 4.8, we observe that the p-values are smaller than 0.05 for both GD and S criteria on all groups. Considering a commonly used significance level of 0.05, this t-test indicates that there are significant differences regarding the performances between HPSO and NSGA-II. Tests on the other criteria show consistent results.

n	HPSO		NSGA-II		p-value	Significance
	$Mean$	SD	$Mean$	SD		
6	0.025	0.009	0.065	0.037	0.046	Yes
8	0.021	0.013	0.063	0.042	0.039	Yes
10	0.020	0.011	0.068	0.053	0.033	Yes
30	0.035	0.017	0.108	0.067	0.030	Yes
50	0.025	0.015	0.223	0.085	0.028	Yes
100	0.028	0.018	0.255	0.121	0.025	Yes

Table 4.7: t-test results of HPSO and NSGA-II with respect to the GD criterion

n	HPSO		NSGA-II		p-value	Significance
	$Mean$	SD	$Mean$	SD		
6	66.78	7.42	79.06	18.33	0.042	Yes
8	68.75	5.98	100.14	27.95	0.041	Yes
10	81.32	12.87	199.66	32.64	0.037	Yes
30	64.07	10.20	98.02	25.43	0.034	Yes
50	54.29	16.85	107.86	22.57	0.030	Yes
100	55.83	13.29	195.47	37.28	0.023	Yes

Table 4.8: t-test results of HPSO and NSGA-II with respect to the S criterion

4.5.5 Effectiveness of the selection strategies for *pbest* and *gbest*

In traditional multi-objective particle swarm optimization algorithms, after each generation, the selection of *gbest* is usually based on density measure which indicates the aggregating degree of the particles within the swarm. The most representative density measures used in the area of multi-objective optimization are nearest neighbor density estimator[57] and

[57] Cf. Deb et al. (2002): *NSGA-II.*

kernel density estimator.[58] In our proposed HPSO, the selection strategy of *pbest* complies with most multi-objective particle swarm optimization algorithms. While for *gbest*, we use a simple and effective strategy: Random selection of one solution from the archive set for each particle.

It is important to investigate whether or not our selection strategies ensure sufficient diversity. Therefore, we apply HPSO and HPSO-ND on all of the 360 instances. HPSO-ND is a modified version of HPSO, whereby the selection of *gbest* is not done randomly but by the nearest neighbour density estimator which has wide applications.[59] Both, HPSO and HPSO-ND, show consistent and similar performance on criteria $NNDS, GD, C$, and S. Remarkable difference is only observed with CT. For better illustration, the comparative results with respect to $S(PF)$ and CT are reported in Table 4.9.

n	HPSO		HPSO-ND	
	$S(PF)$	CT	$S(PF)$	CT
6	66.78	58.45	67.33	65.80
8	68.75	93.48	68.71	107.94
10	81.32	143.01	81.30	169.36
30	64.07	273.69	64.12	311.45
50	54.29	414.74	54.24	493.53
100	55.83	820.85	55.82	965.66
Average	65.26	**300.70**	65.25	**352.29**

Table 4.9: The comparison between HPSO and HPSO-ND with respect to the S and CT criteria

It can be seen from Table 4.9 that HPSO-ND obtains a slightly smaller average value of $S(PF)$ and a considerably larger average value of CT (marked in bold) compared to HPSO. It suggests that the diversity of HPSO-ND leads to no noticeable improvement of the results but requires approximately one sixth more computational efforts due to the sophisticated nearest neighbour density estimator. This demonstrates the effectiveness of the simple selection strategies used in our HPSO.

4.5.6 Analysis of embedded tabu search

In this subsection, the effectiveness of the embedded tabu search procedure (TS) is analysed. The tabu search procedure is to iteratively improve the solution in the neighbourhoods by preventing the reversal of recent moves with short-term memory, which plays an important role in the proposed HPSO algorithm.

[58]Cf. Goldberg / Richardson (1987): *Genetic algorithms with sharing*.
[59]See e.g. Deb et al. (2002): *NSGA-II*.

In this context, we conduct experiments on benchmark instances with up to 10 jobs to compare HPSO with its variant HPSO-LS. HPSO-LS is a modified version of HPSO where the tabu search procedure is replaced by a local search (LS). Similar to the TS procedure, the LS procedure sequentially operates on the IN, SW, and SP neighborhoods generated by insert, swap, and speed moves, respectively. Furthermore, the LS uses $first$ improvement strategy which stops at each generation of HPSO when the first dominating or non-dominated solution is found. Table 4.10 reports the comparison results of HPSO and HPSO-LS with respect to the five comparison criteria $NNDS$, S, C, GD, and CT.

n	m	m_k	PF		HPSO					HPSO-LS				
			$NNDS$	S(PF)	$NNDS$	S(HPSO)	C(HPSO, HPSO-LS)	GD(HPSO, PF)	CT	$NNDS$	S(HPSO-LS)	C(HPSO-LS, HPSO)	GD(HPSO-LS, PF)	CT
6	2	2	62	65.54	62	65.54	1	0.02	15.32	39	69.48	0.93	0.09	45.35
		3	62	62.34	62	62.34	1	0.02	22.89	37	113.39	0.95	0.11	52.67
	4	2	75	70.79	74	71.93	1	0.01	37.6	63	78.59	0.88	0.09	62.34
		3	72	66.66	71	67.32	1	0.01	58.45	63	69.85	0.92	0.1	80.2
8	2	2	85	54.68	85	54.68	1	0.01	84.37	43	91.34	0.92	0.1	97.18
		3	64	71.5	64	71.5	1	0.02	88.54	41	93.03	0.93	0.11	112.24
	4	2	77	68.91	77	69.27	1	0.01	105.62	69	79.93	0.62	0.11	125.67
		3	83	73.87	75	79.54	1	0.01	95.4	72	63.76	0.72	0.07	168.47
10	2	2	81	52.68	80	53.27	1	0.01	114.2	51	64.93	0.85	0.11	192.5
		3	82	68.63	82	68.63	1	0.01	127.56	54	81	0.87	0.1	206.38
	4	2	85	95.62	72	109.46	1	0.01	152.36	81	55.69	0.38	0.06	297.26
		3	88	82.31	75	93.93	1	0.02	177.9	78	58.12	0.47	0.06	312.48
Average			76	69.46	**73.25**	**72.28**	**1**	**0.01**	**90.02**	58	76.59	0.79	0.09	146.06

Table 4.10: The comparison results of HPSO and HPSO-LS on the benchmark instances

It can be found from Table 4.10 that HPSO performs considerably better in terms of non-dominated solutions and coverage ($NNDS$, C) compared to HPSO-LS. Moreover, the average values of S, GD and CT obtained by HPSO are smaller than those of HPSO-LS. Although TS requires more computational efforts than LS to record and check the tabu status at each iteration, the total computation time is lower. This might be due to the fact that TS needs a smaller number of iterations to obtain the pareto front solutions than LS.

In order to visualize the performance of HPSO, HPSO-LS, and NSGA-II, we apply one run of each algorithm on four instances of different sizes, and plot the non-dominated solutions in Fig. 4.7. As shown in Fig. 4.7, the proposed HPSO algorithm is able to obtain better solutions than HPSO-LS and NSGA-II in terms of solution quality and distribution. Comparing NSGA-II and HPSO-LS, we observe that the curves of NSGA-II are further deviated from the fronts of HPSO while HPSO-LS obtains competitive results. It may be explained that NSGA-II solely relies on the population structure, while a local search

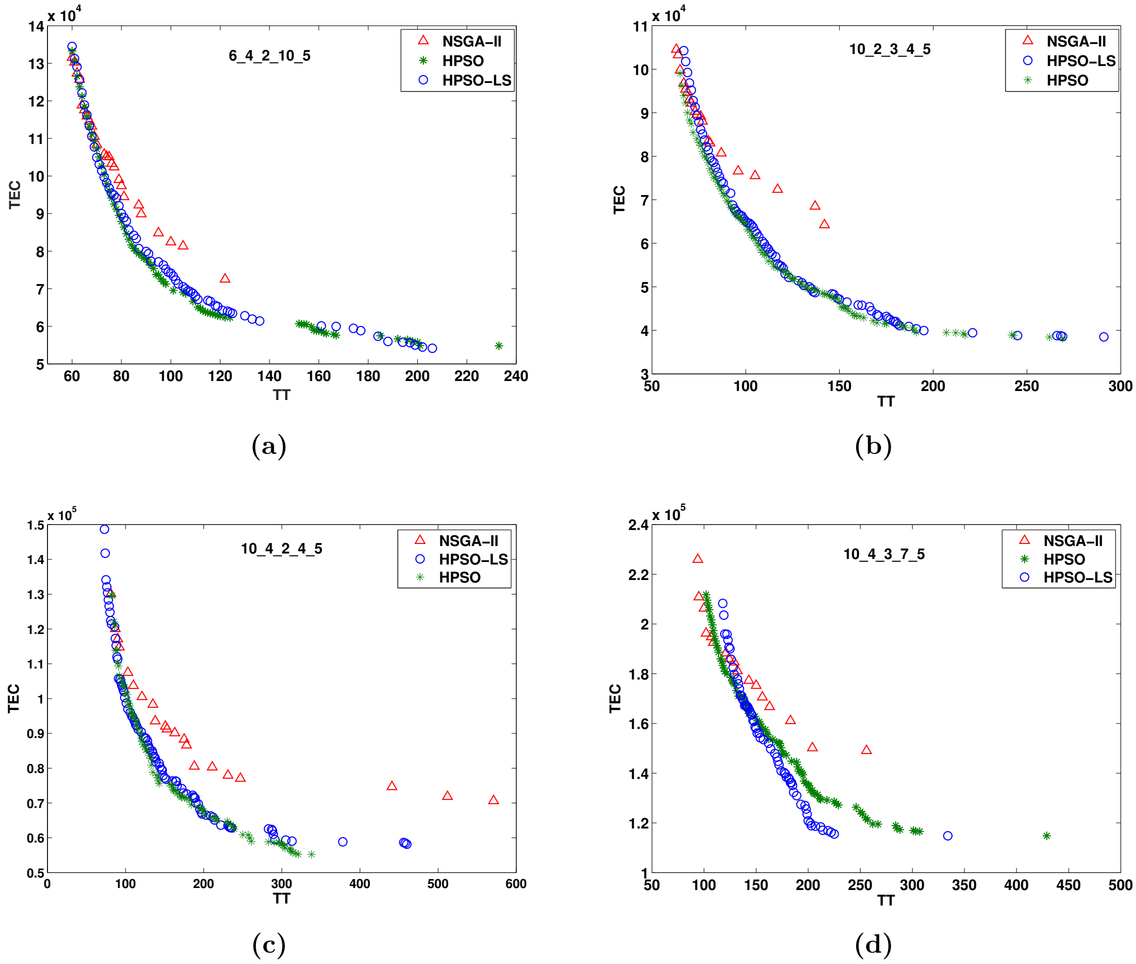


Figure 4.7: The distribution of the non-dominated solutions obtained by different algorithms

strengthens HPSO-LS. These figures illustrate and confirm the results derived from the numerical analysis based on the performance criteria.

4.5.7 Comparison among different PSO-based methods

Our HPSO is a specific hybridization of the multi-objective PSO[60] and TS, which combines the global diversification abilities of a population and the local intensification abilities of local search methods. To further test the performance of HPSO, we compare HPSO with several PSO-based methods: MOPSO and HPSO-SA in addition to HPSO-LS. In detail, MOPSO is the traditional multi-objective PSO, which is a variant of HPSO by removing the the TS procedure. HPSO-SA is a variant of HPSO by replacing TS with simulated

[60]Cf. Reyes-Sierra / Coello (2006): *Multi-Objective Particle Swarm Optimizers.*

annealing procedure.

The simulated annealing in HPSO-SA uses a hill-climbing criteria in order to escape the local minimum. Given a current solution X_c, a new candidate solution X is selected from the neighbourhood and compared with the current solution according to equation (4.33). If X_c is dominated by X, then X is accepted as the new solution. Otherwise, there is a chance for accepting X as a new solution with probability P_{SA} that depends on the difference in their objective function values:

$$P_{SA} = \min\left\{1, \prod_{a=1}^{2} \exp \frac{-[f_a(X) - f_a(X_c)]^+}{Te_{ab}}\right\} \tag{4.33}$$

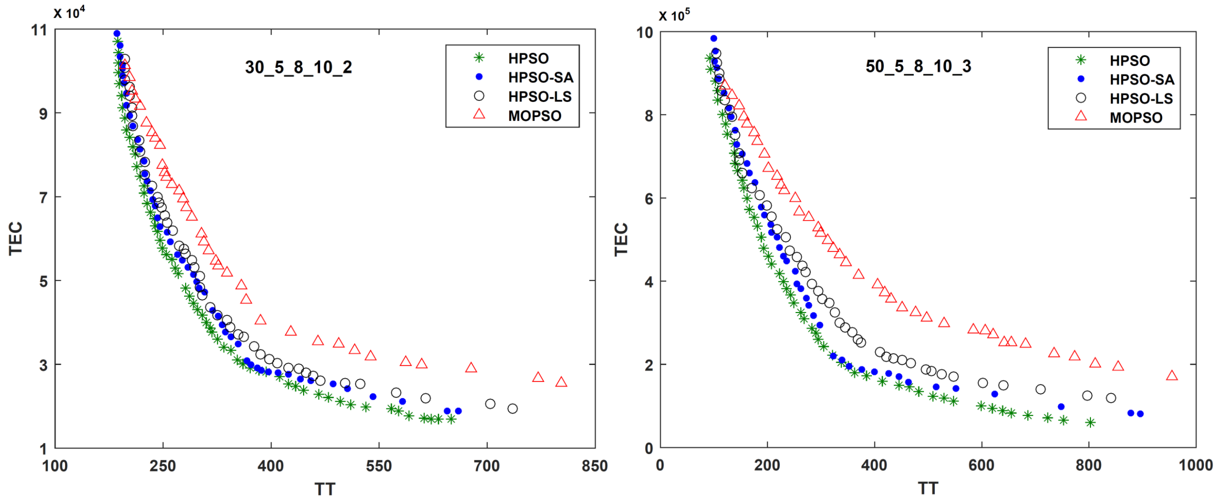
$$[f_a(X) - f_a(X_c)]^+ = \begin{cases} 0, & \text{if } f_a(X) < f_a(X_c) \\ f_a(X) - f_a(X_c), & \text{else} \end{cases} \tag{4.34}$$

where $f_a(X)$ is the a-th objective function value, Te_{ab} is the temperature parameter for the a-th objective at iteration b, which decreases with $Te_{a(b+1)} = \beta Te_{ab}$. In our experiments, the initial temperature $Te_{10} = 10000$ for objective TEC, $Te_{20} = 100$ for objective TT, and $\beta = 0.96$ are adopted.

n	HPSO		HPSO-SA		HPSO-LS		MOPSO	
	$S(HPSO)$	$GD(HPSO, PF)$	$S(HPSO-SA)$	$GD(HPSO-SA, PF)$	$S(HPSO-LS)$	$GD(HPSO-LS, PF)$	$S(MOPSO)$	$GD(MOPSO, PF)$
6	66.78	0.025	73.92	0.039	82.83	0.098	85.70	0.103
8	68.75	0.020	65.34	0.043	82.02	0.098	90.21	0.119
10	81.32	0.022	76.01	0.030	64.94	0.083	87.33	0.107
30	64.07	0.035	70.44	0.036	71.30	0.076	83.98	0.120
50	54.29	0.025	63.78	0.047	66.82	0.081	96.43	0.168
100	55.83	0.028	66.39	0.032	69.41	0.091	92.50	0.154
Average	**65.17**	**0.026**	69.31	0.038	72.89	0.088	89.36	0.129

Table 4.11: The comparison among different PSO-based algorithms with respect to the S and GD criteria

To evaluate the performance of different PSO-based methods, we run HPSO, HPSO-SA, HPSO-LS, and MOPSO on all the 360 instances and report the results in Table 4.11. It can be observed from Table 4.11 that HPSO achieves the best performance with the smallest values of S and GD criteria. Besides, HPSO-SA outperforms HPSO-LS and MOPSO. Results on measures $NNDS$ and C are consistent with S and GD. CT requirements are similar for HPSO, HPSO-LS, and HPSO-SA. MOPSO needs slightly less computing time (about 82% of the computing time taken by HPSO on average) due to a simplified



(a) (b)

Figure 4.8: The comparison among different PSO-based algorithms

algorithm structure without local search. On the other hand, the additional amount of computing time of HPSO, HPSO-LS, and HPSO-SA is justified by the higher solution quality. This comparison indicates the necessity and effectiveness of the search methods embedded in the particle swarm optimization framework.

Furthermore, for visualisation purposes, we plot the pareto fronts obtained by HPSO, HPSO-SA, HPSO-LS, and MOPSO on instances 30_5_8_10_2 and 50_5_8_10_3 in Fig. 4.8. It confirms that the pareto front of HPSO dominates those of the other three methods, and HPSO-SA generally outperforms HPSO-LS and MOPSO. MOPSO, without any local improvement methods, shows the worst performance among all of them. This illustrates the superiority by combining PSO and TS algorithm for solving the considered problem.

4.6 Conclusion

This paper presents a hybrid particle swarm optimization (HPSO) algorithm for the multi-objective optimization of a flexible flow shop scheduling problem, where the total tardiness and the electric power cost at the presence of time-of-use electricity prices are to be minimized simultaneously. In order to adopt HPSO for this discrete optimization problem, particles are represented based on job operation and machine assignment, which are updated directly in the discrete domain. Besides, we propose a multi-objective tabu search procedure to optimize the particles. An assignment/sequence based crossover operator is also used to update the positions of the particles. Experiments are performed on benchmark instances to investigate the quality of the proposed algorithm against the well-known algorithms in the literature, where the results indicate the suitability of HPSO in terms of number of non-dominated solutions, computational time as well as solution quality.

There are several potential directions for future research. First, the HPSO algorithm can be extended to optimize other objectives for the considered problem such as total completion time, machine workload, etc. It is also interesting to design more powerful and sophisticated neighbourhoods based on the basic moves to further intensify the search. Other nature inspired metaheuristics such as ant colony optimization and artificial bee colony algorithm are also worthy to be employed for FFSP by considering problem-specific knowledge. Similar to other scheduling problem with energy considerations, more practical settings, for instance, different execution modes could be introduced. In addition to energy consumption, a direct consequence of pollution emission can also be integrated in the formulation, either as another objective or as an important constraint.

5 A multi-criteria MILP formulation for energy aware hybrid flow shop scheduling

Abstract

This paper introduces an energy aware hybrid flow shop problem considering variable discrete production speeds. In recent years different papers were published dealing with energy aware scheduling. Overall, three different approaches can be identified. In detail, the energy consumption can be reduced by specific planning, time-dependent electricity cost might be exploited or peak power may be decreased. In contrast to the majority of energy aware scheduling models these ideas are adopted simultaneously in the proposed extensive MILP formulation. With this, interdependencies of the different strategies (especially contrary effects of peak power minimization and demand charge reduction) can be considered.

Acknowledgement

Published Paper: S. Schulz (2018): A Multi-criteria MILP Formulation for Energy Aware Hybrid Flow Shop Scheduling. In: *Operations Research Proceedings 2016.* Springer, pp. 543–549.

5.1 Introduction

To reduce electricity demand, companies normally invest in new technologies and processes. However, with intelligent scheduling we are also able to reduce energy demand and costs without losing productivity. Moreover, scheduling has two major advantages: firstly, no high investments are necessary and secondly, it can be implemented immediately.

There are three different strategies in EAS which can be pursued to reduce energy costs. *Reducing energy consumption* directly is the first approach. Such savings can be achieved by selecting parallel machines with low energy consumption, by decreasing production speed or by taking advantage of different machine states like idle or standby (intelligent on/off decisions). A second strategy is to *make use of time depending energy prices*. By shifting energy consumption from peak price times to times of lower energy prices, energy costs can be reduced while energy consumption stays at the same level. Besides the consumption charge, companies often pay also a demand charge for the maximum power demand during the billing period. A third approach in EAS is now to level the energy needs in order to *lower the peak power* and hence the demand charge.

Only a handful hybrid flow shop problems consider some of the mentioned approaches. In Bruzzone et al. (2012)[1] an EAS problem can be found, whereby the peak power is gradually reduced on the basis of an APS-system. Whereas Luo et al. (2013)[2] publish a hybrid flow shop problem with variable machine speed and time depending electricity prices, an approach for on-off decisions for a closed loop flow shop plant is presented in Mashaei / Lennartson (2013)[3]. Also Dai et al. (2013)[4] consider different machine states in a flexible flow shop problem to reduce energy consumption and makespan simultaneously. Tan / Huang / Liu (2013)[5] describe a two-stage mathematical programming approach to solve a parallel hybrid scheduling problem in steel making process with variable energy prices.

To the best of our knowledge, there is no paper considering the three mentioned basic strategies simultaneously. The primary concern of this paper is to analyze the effects and interdependencies of different EAS measures. Therefore, a comprehensive MIP including a wide range of energy-aware aspects is developed in section 5.2. In section 5.3 a numerical example serves to illustrate how the model operates as well as to visualize the interdependencies in EAS. Section 5.4 gives a short summary.

[1] Bruzzone et al. (2012): *Energy-aware scheduling.*
[2] Luo et al. (2013): *Hybrid flow shop scheduling.*
[3] Mashaei / Lennartson (2013): *Energy Reduction in a Pallet-Constrained Flow Shop.*
[4] Dai et al. (2013): *Energy-efficient scheduling.*
[5] Tan / Huang / Liu (2013): *Scheduling Under Variable Electricity Price.*

5.2 A comprehensive MIP for EAS

Indices		**Parameters**			
j	Job in J	P_{max}	Maximum peak power	ce_t	Electricity cost
m	Machine in M_s	E_{sj}^m	Energy consumption	cp_j	Production cost
s	Production stage in S	D_j	Due date	ct_j	Tardiness cost
t	Time period in T	R_j	Release date	a_s^{mv}	Energy savings
v	Speed level in V	S_{sj}^m	Standard processing time		

Decision Variables

$c_{sj} \in \mathbb{N}$	Completion time of task s of job j
$g_{sj}^{mv} \in \{0,1\}$	Processing time extension v of task s of job j on machine m
$T_j \in \mathbb{N}$	Tardiness of job j
$p_{sjt}^m \in \mathbb{N}$	Power consumption of task s of job j on machine m in time period t
$x_{sjt}^m \in \{0,1\}$	Task s of job j is performed on machine m in time period t
$y_{sj}^m \in \{0,1\}$	Task s of job j is assigned to machine m
$z_{sjt}^m \in \{0,1\}$	Execution of task s of job j on machine m starts in time period t

Every job j has to be processed at each production stage and in every stage s there is a set of unrelated parallel machines denoted as M_s. The planning horizon is divided into T_{max} time-intervals of equal length. Using the introduced notations above the EAS Mixed Integer Problem can be modelled as follows:

Minimize

$$\sum_{j\in J}(ct_j \cdot T_j + cp_j \cdot (c_{S_{max}j} - (R_j - 1))) + \sum_{t\in T}(ce_t \cdot \sum_{j\in J}\sum_{s\in S}\sum_{m\in M_s} p_{sjt}^m) \tag{5.1}$$

Subject to:

$$\sum_{j\in J} x_{sjt}^m \leq 1 \quad \forall\, s, m, t \tag{5.2}$$

$$\sum_{t\in T}\sum_{m\in M_s} z_{sjt}^m = 1 \quad \forall\, j, s \tag{5.3}$$

$$x_{sjt}^m \leq z_{sjt}^m \quad \forall\, j, s, m, t = R_j \tag{5.4}$$

$$x_{sjt}^m - x_{sj,t-1}^m \leq z_{sjt}^m \quad \forall\, j, s, m, t \geq R_j \tag{5.5}$$

$$\sum_{t\in T} x_{sjt}^m = S_{sj}^m \cdot y_{sj}^m + \sum_{v\in V} g_{sj}^{mv} \quad \forall\, j, s, m \tag{5.6}$$

$$\sum_{m\in M_s} y_{sj}^m = 1 \quad \forall\, j, s \tag{5.7}$$

$$\sum_{t\in T|t\geq R_j}\sum_{m\in M_s}(z^m_{sjt}-z^m_{s-1,jt})\cdot t\geq\sum_{m\in M_{s-1}}(S^m_{s-1,j}\cdot y^m_{s-1,j}+\sum_{v\in V}g^{mv}_{sj})\quad\forall\, j,s>1 \tag{5.8}$$

$$\sum_{t\in T|t\geq R_j}\sum_{m\in M_s}z^m_{sjt}\cdot t=c_{sj}-\sum_{m\in M_s}(S^m_{sj}\cdot y^m_{sj}+\sum_{v\in V}g^{mv}_{sj})+1\quad\forall\, j,s \tag{5.9}$$

$$T_j\geq c_{sj}-D_j\quad\forall\, j,s \tag{5.10}$$

$$p^m_{sjt}=\max(0;E^m_{sj}\cdot(x^m_{sjt}-\sum_{v\in V}g^{mv}_{sj}\cdot a^{mv}_{sj}))\quad\forall\, j,s,m,t \tag{5.11}$$

$$\sum_{v\in V}g^{mv}_{sj}\leq S^m_{sj}\cdot y^m_{sj}\quad\forall\, j,s,m \tag{5.12}$$

$$g^{m,v-1}_{sj}\geq g^{mv}_{sj}\quad\forall\, j,s,m,v>1 \tag{5.13}$$

$$\sum_{j\in J}\sum_{s\in S}\sum_{m\in M_s}p^m_{sjt}\leq P_{max}\quad\forall\, t \tag{5.14}$$

(5.2) ensures that every machine can process only one job in each period. Since non-preemption is assumed, (5.3) guarantees that each job has only one starting time period at each production stage. (5.4) and (5.5) are necessary to determine z^m_{sjt} depending on x^m_{sjt}. We introduce (5.6) to accurately reflect processing time that consists of standard processing time and extra time caused by production speed reductions (g^{mv}_{sj}). Hereby, y^m_{sj} serves to select machine m for each job at production stage s. With (5.7) every job is exactly allocated to one machine at each stage. As a matter of course, no job can be scheduled on a machine before the previous job on this machine is completed (5.8). (5.9) serves to calculate the completion time of a task, the tardiness of a job finally results from (5.10).

Since it is assumed that the parallel machines have different energy efficiencies for different jobs, energy consumption can be reduced by assigning jobs to machines with lower demand. Energy demand may also be reduced by decreasing production speed. Therefore, in (5.6) and (5.8) the possibility of increasing the manufacturing time gradually is already considered by g^{mv}_{sj}. Depending on the additional time energy consumption is reduced by the percentage a^{mv}_{sj} in (5.11).

While energy conversion efficiency is very high for power usage greater than 75% of rated load, electric motors operating slower than 50% of maximal speed show excessive wear and energy consumption in relation to the production output.[6] Condition (5.12) ensures that the cumulative number of speed reductions can never exceed the standard processing time and a throttling higher than 50% is thus avoided. Furthermore, (5.13) is deployed in order to enable stepwise speed changes while making sure that no speed level is skipped.

An important characteristic of the model is to take advantage of energy price fluctuations.

[6] Cf. Kaya et al. (2008): *Energy efficiency in pumps.*

This unavoidably requires a time-dependent electricity price ce_t. Besides the electricity costs, the objective function (5.1) minimizes delays and total completion time which are multiplied by a cost factor. Since energy costs consist of consumption and demand charges also costs for energy peak should be considered.

Peak load charges have to be paid for long time periods (quarterly, yearly). In contrast, scheduling is used most commonly for the purpose of operational decision-making and it normally examines shorter periods (daily, weekly). Therefore, optimizing peak load costs within the scheduling model is only advisable if the billing period corresponds to the period considered. The approaches of Babu / Ashok (2008)[7] or Fang et al. (2011)[8] are examples of models that directly include energy peak costs in the objective function.

In this contribution a different approach is pursued. Due to the usually unequal time periods of peak load charges and scheduling horizon it is preferred to integrate the peak load as a constraint into our model. Often the maximum peak power is known from the past. Constraint (5.14) ensures that energy consumption is always lower than this value. By varying P_{max} a Pareto front can be developed and the peak load can be improved too.

5.3 Computational experiments

A two-stage hybrid flow shop with two parallel machines on each stage shall serve as an example. Eight jobs with non-identical release and due dates are considered. All examples are based on randomly generated parameters within given ranges.

In order to give a high incentive to meet due dates, the tardiness cost parameter ct_j is put at 500 for each job j. Obviously, the ratio between production and energy costs is of substantial importance. It is assumed that 50 % of the variable costs are energy costs. To allow the energy costs to be as realistic as possible, Phelix spot market prices (15. August 2015) are used. The prices are depicted in Figure 5.1. Considering the average energy consumption, the production cost factor is assumed to be 100. Additionally, the energy savings depending on production speed reductions are required. For the example electric motor energy savings following Saidur et al. (2009)[9] are discretized. The EAS-model is solved using IBM ILOG CPLEX. All problem instances are tested on an Intel Xeon, 3.46 GHz computer.

Leaving aside (5.14) leads to an energy peak of 45. Based on this value P_{max} will be parametrically reduced. To keep the calculation time low, the optimal costs of the previous

[7] Babu / Ashok (2008): *Peak load management.*
[8] Fang et al. (2011): *A new approach to scheduling.*
[9] Saidur et al. (2009): *Energy and emission analysis.*

P_{max}	45	39	37	36	35	30	25	20	15	11
Total Cost	17584.8	17776.1	17849.5	18089.6	18265.9	18358.3	20282.2	22040.2	25095	32031.4
Energy Cost	8384.8	8576.1	8749.5	8889.6	9165.9	8558.3	8282.2	7240.2	7195	6031.4
Delays	3	3	3	3	3	4	7	11	16	29
E. demand	386.5	370.6	385.7	376.1	395.5	368.5	344.7	302.2	280.0	228.4
Makespan	19	19	19	19	19	19	21	22	21	24
Throttlings	5	5	6	5	6	5	10	15	17	24

Table 5.1: Selected solutions of the numeric example

instance are always the lower bound for the next lower peak power scenario. Selected parts of the results are represented in table 5.1.

At first, P_{max} can be reduced without any changes in the results. After this initial reduction total costs increase with lower peak power. Therefore, it must be kept in mind that peak power charges decline with lower peak power and these charges are not included in the total costs. Moreover, a diminishing maximum peak power goes along with decreasing energy demand, while makespan, delays and computing time increase. The energy costs tend to decrease with shrinking maximum peak power. Nevertheless, due to the volatile energy prices it is possible that lower energy demand leads to higher energy costs.

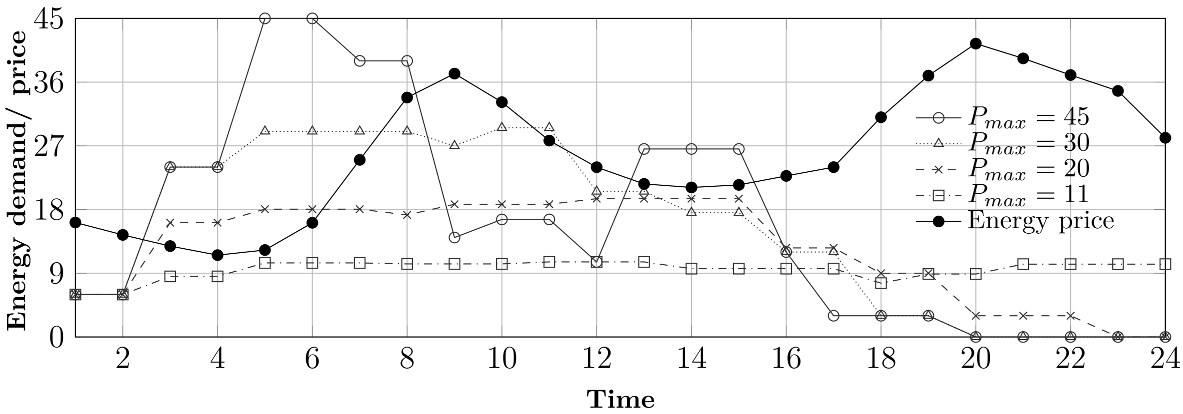


Figure 5.1: Energy consumption for different peak power scenarios and RTP price

Peak power reduction causes postponements and more production speed throttling and hence lower total energy demand. The load curves in Figure 5.1 illustrate the effects on energy demand. By taking a closer look at the curves it can be noted, that the possibilities of taking advantage of energy price fluctuations decrease with lower peak power. This is due to the leveling effect on the energy consumption that goes along with lower P_{max}-values. The example illustrates what theoretically has already been explained. Reducing energy peak and making use of time depending energy prices are contrary objectives (energy cost dilemma).

Besides this insight also the general influence of energy cost consideration and variable production speed shall be examined. Therefore, our EAS-model will be solved further three times disregarding certain aspects. All scenarios are put into relation with the basic model as regards our cost-oriented objective function and the energy consumption. The results are shown in Figure 5.2.

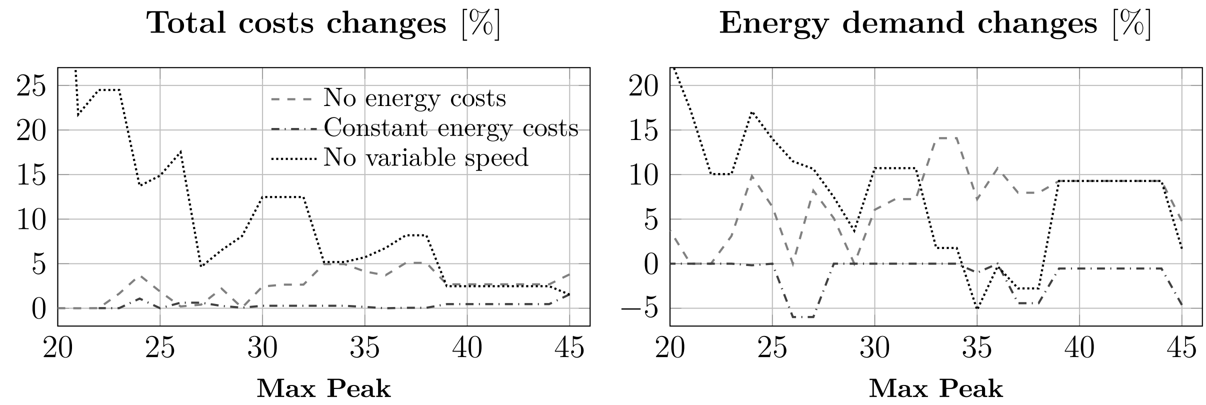


Figure 5.2: Cost and energy demand changes with problem variations

As might be expected, ignoring some aspects leads to higher total costs in all scenarios. The influence of time-depending energy prices is relatively low. For $P_{max} = 45$ the costs are 1.6 % higher, but for lower peak power the gap is less than 1 %. Interestingly, the scenario with constant energy prices leads to less energy consumption.

Without any energy price considerations costs increase up to 5 % and energy consumption up to 14 %. This means that the consideration of energy costs can lead to significant savings. Large percentage deviations can occur, if the production speed is assumed to be fixed. Facing low peak power limits, speed changes are especially important to avoid cost increases up to 25 %.

5.4 Conclusions and further research

In this paper a cost-oriented energy-aware MILP-model was developed to solve hybrid flow-shop problems. Recently, several articles have been published concerning EAS. However, to the best of our knowledge none of them simultaneously examines all three basic strategies of EAS mentioned above. To close this research gap, a comprehensive MILP for hybrid flow shop scheduling is formulated. The specific functioning of the model was illustrated by a numerical example and the impact of different EAS strategies was investigated. It could be shown that there are contrary effects in the different approaches of EAS. Especially peak power reduction and exploiting time depending energy prices are contrary objectives.

6 A multi-objective iterated local search algorithm for comprehensive energy-aware hybrid flow shop scheduling

Abstract

Growing environmental awareness and the relevance of energy costs in many industries has led to the need of improving energy efficiency in operations management; hence, energy-aware scheduling has grown in importance. In EAS three basic strategies can be identified. First, a large part of research activities is aimed at reducing *energy consumption*; second, energy costs can be reduced by making use of *varying energy prices*; third, a rarely-examined aspect is *load curve leveling*, used to reduce demand charges or grid utilization charges. In this paper, all three strategies are integrated into one model for the first time in order to solve a multi-objective hybrid flow shop scheduling problem. A new multiphase iterated local search algorithm (ILS) is developed to determine a three-dimensional Pareto front regarding three objectives: makespan, total energy costs and peak load. Tabu lists, several time- and energy-dependent list scheduling algorithms, a right-shifting procedure and a reference point based fitness function enable high-quality solutions. A computational study is presented that analyzes the interdependencies of objectives and compare the proposed algorithm to well-known NSGA2 heuristic. The ILS is proven to be suitable in purposeful search in the solution space, which allows practical decision support.

Acknowledgement

Published Paper: S. Schulz / J. S. Neufeld / U. Buscher (2019): A multi-objective iterated local search algorithm for comprehensive energy-aware hybrid flow shop scheduling. In: *Journal of Cleaner Production*, vol. 224, pp. 421–434. ISSN: 0959-6526.

6.1 Introduction

Energy is an indispensable factor in present-day production environments, and energy costs form a crucial cost component for many manufacturing companies.[1] In addition to financial incentives, environmental protection efforts and the quest for a sustainable power supply have led to a greater awareness of energy consumption in industry. Thus it is not surprising that EAS - which aims to reduce energy demand or energy costs through an intelligent scheduling of jobs - has recently received increasing attention in research and practice.[2] Compared to technological measures that reduce energy-consumption, EAS has two major advantages: first, no high investments are necessary; second, it can be implemented immediately. At the same time, conventional objectives like makespan or flow time can be integrated into the applied algorithms.

In EAS, three different strategies can be pursued to reduce energy costs.[3] *Reducing consumption* is the first strategy, which can be attained in multiple ways; in heterogeneous parallel machine environments in particular, machines with low energy consumption can be selected. In addition, energy costs can be reduced by decreasing production speed at the expense of productivity. Another possibility is to take advantage of different machine states, such as idle or standby, and to optimize energy consumption by making intelligent on/off decisions.

Another strategy makes use of *varying energy prices.* Although energy consumption remains at the same level, expenses can be reduced by shifting energy consumption to times of lower energy prices. Typical pricing models are Time-of-use (TOU) or Real-Time Pricing (RTP). While TOU energy prices are specified in advance for certain time periods of the day, RTP is based on spot market prices, which (for example) are updated every 15 minutes at the European Power Exchange (*EPEX SPOT SE*).

A final strategy also utilizes the pricing mechanism but is based on *load curve leveling* to reduce the peak power (Critical Peak Pricing). In addition to the consumption charge, companies normally pay a so-called demand charge for the maximum power peak during the billing period.[4] Energy needs can be leveled in order to lower the maximum power peak and thus the demand charge. While the peak power fee normally covers a time period of at least several weeks, scheduling is primarily dedicated to shorter production periods; therefore, directly integrating the demand charge into a scheduling problem is not easy. However, since the demand charge per kW is 200 to 400 times higher than the electricity

[1] Cf. Abdelaziz / Saidur / Mekhilef (2011): *A review on energy saving strategies.*
[2] Cf. Yan et al. (2016): *Energy-efficient flexible flow shop scheduling.*
[3] Cf. Schulz (2018): *A Multi-criteria MILP Formulation.*
[4] Cf. Bego / Li / Sun (2014): *Identification of reservation capacity in critical peak pricing.*

price per kWh, reducing the energy peak can be fairly lucrative.[5]

Since EAS integrates one or several of these strategies into conventional scheduling models, multi-objective optimization approaches are needed to take into account the different requirements and goals of practical problems. For the first time, in this paper all three strategies are integrated into an energy-aware multi-objective hybrid flow shop scheduling model: unrelated parallel machines in the hybrid flow shop enable a reduction of energy consumption, while RTP prices are utilized to model varying energy charges. In addition to total energy costs, maximum peak power and makespan are considered as additional objectives. In particular, the applied Pareto optimization enables viable decision support as the decision maker can choose between several solutions according to individual preferences regarding the considered criteria. Furthermore, Pareto optimization allows for in-depth analysis of trade-offs and interaction between objectives, which in turn allows for a broader understanding of the studied problem.[6] While population based metaheuristics are usually applied for multi-objective optimization and EAS, we propose a new iterated local search algorithm that allows the of use several problem-specific list scheduling mechanisms. The idea of reference points[7] is adapted to adjust the objective function before each start of the local search procedure, similar to Zhang / Li (2007)[8]. Furthermore, an efficient perturbation strategy is chosen.

The remainder of the paper is structured as follows. Section 6.2 gives a short overview of the relevant literature on EAS and points out current research gaps, followed by the formal problem definition in Section 6.3. The ILS approach is explained in Section 6.4. In Section 6.5 a numerical example serves to illustrate the effects and interdependencies of the different objectives. Based on newly generated problem instances the ILS algorithm is compared against NSGA2, which is state-of-the-art for similar multi-objective optimization problems. The extensive computational study in Section 6.6 proves the effectiveness of the proposed algorithm and allows an in-depth analysis of the problem. Section 6.7 concludes with directions for future research.

6.2 Related literature

Significant relevant sources for the presented study include energy-aware scheduling, especially in hybrid flow shops, and (multi-objective) iterated local search algorithms. Energy-aware scheduling has recently received considerable attention. Since Boukas /

[5]Cf. Nghiem et al. (2011): *Green scheduling.*
[6]Cf. Deb (2014): *Search methodologies.*
[7]Cf. Deb / Jain (2012): *Handling many-objective problems.*
[8]Zhang / Li (2007): *A multiobjective evolutionary algorithm.*

Haurie / Soumis (1990)[9] analyzed energy consumption in scheduling for the first time more than 100 papers have been published that cover the three strategies of EAS.[10] Most papers focus on only one or two aspects of integrating energy awareness into various production environments. Often, discussion focuses on reducing energy consumption based on the variation of machine states and production speed[11] or on parallel machines with different consumptions[12]. For varying energy prices, time-of-use prices are mostly considered[13], real-time prices less frequently[14]. Peak power reduction has thus far received only scant attention: usually, it is either integrated into a cost function[15] or as a constraint with a given peak power limit[16].

Hybrid or flexible flow shops are very common in industry[17] and various practical applications have been reported.[18] Nevertheless, only a few EAS approaches have been proposed and discussed in these environments. Dai et al. (2013)[19] consider three different machine states in a flexible flow shop problem, minimizing total energy costs and makespan using genetic algorithm and simulated annealing. Luo et al. (2013)[20] discuss a hybrid flow shop problem with variable machine speed and TOU electricity prices. Based on a preceding optimization of cutting parameters, Yan et al. (2016)[21] simultaneously minimize makespan and total energy consumption by a genetic algorithm for a flexible flow shop. Only Bruzzone et al. (2012)[22] and Xu / Weng / Fujimura (2014)[23] include a limited peak power for this environment into a MILP.

Regardless of the considered shop floor model, to the best of our knowledge only Ashok

[9] Boukas / Haurie / Soumis (1990): *Hierarchical approach to steel production scheduling.*

[10] For a detailed overview of energy-aware scheduling approaches see Gahm et al. (2016): *Energy-efficient scheduling*; Biel / Glock (2016): *Energy-efficient production planning.*

[11] See e.g. Fang / Lin (2013): *Parallel-machine scheduling*; Liu et al. (2014b): *Minimising total energy consumption*; Mansouri / Aktas (2016): *Minimizing energy consumption and makespan.*

[12] See e.g. Ji / Wang / Lee (2013): *Minimizing resource consumption*; Dai et al. (2015): *Energy-aware integrated process planning and scheduling.*

[13] See e.g. Castro / Harjunkoski / Grossmann (2011): *Optimal scheduling of continuous plants*; Che / Zhang / Wu (2017): *Energy-conscious unrelated parallel machine scheduling.*

[14] See e.g. Yusta / Torres / Khodr (2010): *Machining process scheduling in spot electricity markets.*

[15] See e.g. Babu / Ashok (2008): *Peak load management*; Sun et al. (2014): *Inventory control for peak electricity demand reduction.*

[16] See e.g. Bruzzone et al. (2012): *Energy-aware scheduling*; Fang et al. (2013): *Flow shop scheduling with peak power*; Zheng / Wang (2015): *Reduction of carbon emissions.*

[17] Cf. Neufeld / Gupta / Buscher (2015): *A comprehensive review of group scheduling.*

[18] For a detailed classification and overview of solution methods see Ruiz / Vazquez-Rodriguez (2010): *The hybrid flow shop scheduling problem*; Ribas / Leisten / Framinan (2010): *Review and classification of hybrid flow shop scheduling.*

[19] Dai et al. (2013): *Energy-efficient scheduling.*

[20] Luo et al. (2013): *Hybrid flow shop scheduling.*

[21] Yan et al. (2016): *Energy-efficient flexible flow shop scheduling.*

[22] Bruzzone et al. (2012): *Energy-aware scheduling.*

[23] Xu / Weng / Fujimura (2014): *Energy-Efficient Scheduling.*

(2006)[24] and SCHULZ (2018)[25] have integrated the reduction of energy consumption, the use of price volatility and peak power reduction into one model. While SCHULZ (2018) presents a MILP for a similar problem discussed in this paper, ASHOK (2006) describes a model with variable electricity charges, production speed variations and demand charges, in particular critical peak pricing. Since his objective function minimizes monthly operating costs this model is only useful to a limited extent for general scheduling problems. Usually, scheduling models are used to optimize short-term processes, and as a result, long-term charges for the peak load are difficult to integrate into a single objective function. Neither paper considers multiple objectives and therefore show a lack of Pareto optimization.

Moreover, it can be seen that population-based metaheuristics are almost exclusively used for energy-aware scheduling problems. However, since FRAMINAN / LEISTEN (2008)[26] iterated local search has also been applied successfully to multi-objective scheduling problems, particularly in flow shop environments.[27] ILS is closely related to other explorative local search methods, such as iterated greedy and GRASP.[28] GEIGER (2011)[29] proposes a promising Pareto iterated local search that combines iterated local search with variable neighborhood search minimizing makespan and total tardiness for a flow shop problem. Similarly, DUBOIS-LACOSTE / LOPEZ-IBANEZ / STÜTZLE (2011)[30] use a two-phase local search algorithm, and PAN / RUIZ / ALFARO-FERNANDEZ (2017)[31] apply an ILS to hybrid flow shop scheduling with due windows, minimizing earliness and tardiness. Recently, JASZKIEWICZ (2018)[32] has presented a new Pareto Local Search Algorithm that integrates a mechanism for effective exploration of neighborhoods with efficient data structures to update the Pareto front. In its examples, it is applied to traveling salesman problems. We incorporate several ideas from multi-objective ILS algorithms into the studied energy-aware scheduling problem.

[24] ASHOK (2006): *Peak-load management in steel plants.*
[25] SCHULZ (2018): *A Multi-criteria MILP Formulation.*
[26] FRAMINAN / LEISTEN (2008): *A multi-objective iterated greedy search.*
[27] Cf. YENISEY / YAGMAHAN (2014): *Multi-objective permutation flow shop.*
[28] For an overview of the general procedure of ILS see LOURENÇO / MARTIN / STÜTZLE (2003): *Iterated local search.*
[29] GEIGER (2011): *Decision support for multi-objective flow shop scheduling.*
[30] DUBOIS-LACOSTE / LOPEZ-IBANEZ / STÜTZLE (2011): *A hybrid TP+ PLS algorithm.*
[31] PAN / RUIZ / ALFARO-FERNANDEZ (2017): *Iterated search methods.*
[32] JASZKIEWICZ (2018): *Many-Objective Pareto Local Search.*

6.3 Problem definition

6.3.1 Mathematical model

In the following, we consider a hybrid flow shop problem with real time energy prices RTP_t and three different objectives. Overall, J_{max} jobs ($j \in J$) go through S_{max} consecutive production stages ($s \in S$), whereby $M_{s,max}$ parallel heterogeneous machines ($m \in M_s$) are available at stage s. The planning horizon is divided into $t \in T$ time periods of equal length. Each job has a processing time P_{sj}^m and E_{sj}^m is the demand for energy per time period. The problem can be formulated as a mixed integer program using time indexed variables to take time-dependent energy costs into account. The following decision variables are employed:

$p_{sjt}^m \in \mathbb{N}$	Power consumption of task s of job j on machine m in time period t
$x_{sjt}^m \in \{0,1\}$	Task s of job j is performed on machine m in time period t
$y_{sj}^m \in \{0,1\}$	Task s of job j is assigned to machine m
$z_{sjt}^m \in \{0,1\}$	Execution of task s of job j on machine m starts in time period t

Furthermore the model is based on the following fundamental assumptions:

- All jobs have to be processed at 2 stages at least and must follow the same processing sequence. An interruption of a task, i.e. the processing of a job at one stage, is not allowed.
- At least one stage has more than one machine. Parallel machines are assumed to be heterogeneous in terms of energy consumption and processing times.
- All jobs and machines are available at time zero.
- Each machine can process at most one job at a time, and each job can be processed by at most one machine at a time.
- Set-up times and energy consumption during idle times are assumed not to be decision-relevant.

Based on these assupmtions and the introduced nomenclature, the model can be formulated as follows:

$$\sum_{j \in J} x_{sjt}^m \leq 1 \quad \forall\ s, m, t \tag{6.1}$$

$$x_{sjt}^m \leq z_{sjt}^m \quad \forall\ j, s, m, t = 1 \tag{6.2}$$

$$x_{sjt}^m - x_{sjt-1}^m \leq z_{sjt}^m \quad \forall\ j, s, m, t \geq 1 \tag{6.3}$$

$$\sum_{t \in T} \sum_{m \in M_s} z_{sjt}^m = 1 \quad \forall\ j, s \tag{6.4}$$

$$\sum_{t \in T} x_{sjt}^m = P_{sj}^m \cdot y_{sj}^m \quad \forall\, j, s, m \tag{6.5}$$

$$\sum_{m \in M_s} y_{sj}^m = 1 \quad \forall\, j, s \tag{6.6}$$

$$\sum_{t \in T} \sum_{m \in M_s} (z_{sjt}^m - z_{s-1jt}^m) \cdot t \geq \sum_{m \in M_s} P_{s-1j}^m \cdot y_{s-1j}^m \quad \forall\, j, s > 1 \tag{6.7}$$

$$p_{sjt}^m = E_{sj}^m \cdot x_{sjt}^m \quad \forall\, j, s, m, t \tag{6.8}$$

$$\sum_{t \in T} \sum_{m \in M_s} z_{sjt}^m \cdot t = c_{sj} - \sum_{m \in M_s} P_{sj}^m \cdot y_{sj}^m + 1 \quad \forall\, j, s \tag{6.9}$$

$$c_{sj} \geq 0 \quad \forall\, j, s \tag{6.10}$$

$$p_{sjt}^m \geq 0 \quad \forall\, j, s, m, t \tag{6.11}$$

$$x_{sjt}^m \in \{0, 1\} \quad \forall\, j, s, m, t \tag{6.12}$$

$$y_{sj}^m \in \{0, 1\} \quad \forall\, j, s, m \tag{6.13}$$

$$z_{sjt}^m \in \{0, 1\} \quad \forall\, j, s, m, t \tag{6.14}$$

Constraint (6.1) ensures that a machine cannot work simultaneously on different jobs. With (6.2) and (6.3), the starting time is defined, and since each job starts exactly once (see (6.4)), interruptions are excluded. With the introduction of equation (6.5), it is guaranteed that each job is scheduled at each stage for the necessary processing time P_{sj}^m depending on the selected machine. Each job is assigned to one machine at each production stage by (6.6). As a matter of course, no job can be scheduled on a machine before the previous stage of that job is completed (see (6.7)). In (6.8) and (6.9), the energy consumption and the completion time are defined. The variables are specified by (6.10) to (6.14). Based on this model, the objective functions can be formulated as follows:

I. Makespan:
$$C_{max} = \max_{j \in J} \left(c_{S_{max}, j}\right) \tag{6.15}$$

II. Total Energy Costs:
$$TEC = \sum_{j \in J} \sum_{s \in S} \sum_{m \in M_s} \sum_{t \in T} \left(p_{sjt}^m \cdot RTP_t\right) \tag{6.16}$$

III. Peak Power:
$$PP = \max_{t \in T} \left(\sum_{j \in J} \sum_{s \in S} \sum_{m \in M_s} p_{sjt}^m \right) \tag{6.17}$$

As mentioned above, three different objectives are minimized. In addition to the conventional makespan criterion (6.15), two energy cost-related values are considered. Total energy costs (TEC) correspond to the summed up energy demands multiplied by the time-dependent real time prices (6.16). This indirectly minimizes the energy consumption. In most cases, lower energy consumption also reduces energy costs. However, if the planning on a more efficient machine can only take place at times of higher energy costs, and if a machine with higher energy consumption can still be planned at lower energy prices, the opposite can be the case. A large proportion of energy costs is related to the demand charge; thus, reducing peak power consumption (6.17) also plays an important role. In contrast to other publications, peak power is not only considered as an upper bound in a constraint but is explicitly introduced as an objective function, in order to add an additional degree of freedom for the decision maker. Moreover, in practice, the exact determination of a value PP is difficult, with an implicit risk of being overestimated.

Due to the minmax objective functions (6.15) and (6.17), the model is initially non-linear. By introducing $\geq$-restrictions instead of maximization operators, this can easily be linearized, as shown in equations (6.18) and (6.19).

$$C_{max} \geq c_{S_{max},j} \qquad \forall\ j \tag{6.18}$$

$$PP \geq \sum_{j \in J} \sum_{s \in S} \sum_{m \in M_s} p^m_{sjt} \qquad \forall\ t \tag{6.19}$$

However, common solver will handle both formulations. The presented model is implemented in IBM ILOG CPLEX 12.6 and can be solved by branch and cut algorithm for each objective function individually. Possible approaches to combine this problem with the idea of multiple objective functions are discussed in the following.

6.3.2 Illustrative example for decission making with several objectives

The problem of different objectives is most commonly bypassed by blending the criteria or by choosing lexicographic approaches. While blended approaches weight the different objectives and sum them up to one function, the solution in lexicographic models is calculated by considering objectives successively according to their priority. Both methods favorably lead to one optimal solution. An example of the lexicographic concept is given here and illustrates how our model operates.

We consider 10 jobs that have to be processed at one of two parallel machines at two different production stages. Processing time and energy consumption are generated randomly in $unif\{1;10\}$ and can be seen in Table 6.1. For energy costs, we use the RTP from the Phelix spot market on 27th March, 2017. The prices are depicted in Figure 6.1.

For the lexicographic solution, the model is initially solved for the objective with highest priority. Based on the first solution, the model is solved again for the second most-

		Processing Time										Energy Consumption									
Job		**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Stage1	**M1**	6	5	1	3	7	7	2	4	1	4	3	1	3	1	2	4	3	2	5	4
	M2	4	7	3	3	5	6	6	5	3	9	3	8	2	5	9	10	5	6	5	6
Stage2	**M1**	9	8	5	9	8	6	2	5	9	7	1	5	9	6	1	1	7	1	4	3
	M2	7	4	8	3	4	1	9	8	10	6	2	4	8	5	3	6	6	6	10	3

Table 6.1: Numerical example

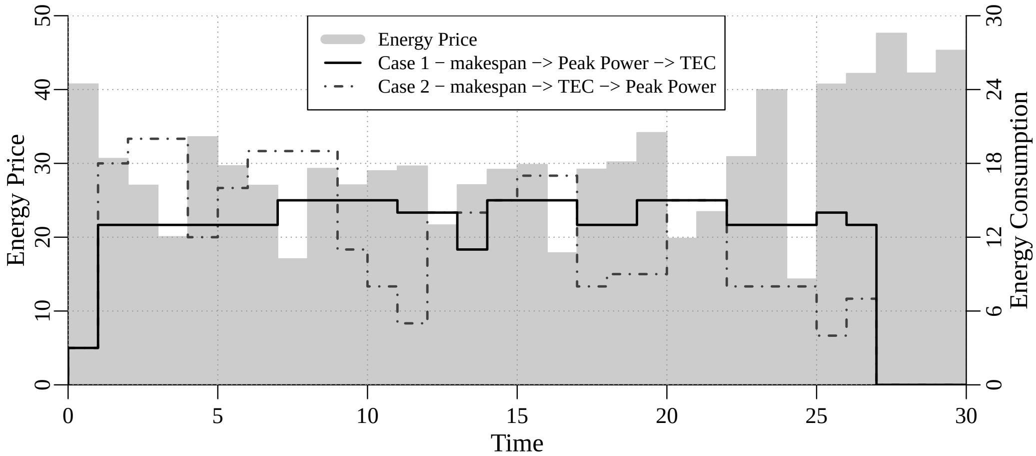


Figure 6.1: Energy price in Euro/MWh and possible load curves for minimal makespan

important object with a fixed value for the most important criterion. The objectives are considered and determined successively according to the priority order, resulting in an iterative procedure.

For the example in Table 6.1, two different lexicographic solutions are generated using the model presented in Section 6.3.1. Figure 6.2 shows the optimal schedule for a priority order: *makespan* $\succ$ *Peak Power* $\succ$ *TEC* (case 1). In this case, the optimal peak power for the minimum makespan of 27 is 15. The corresponding TEC accounts for 10,174.13. Figure 6.3 depicts the solution for priority order: *makespan* $\succ$ *TEC* $\succ$ *Peak Power* (case 2). Here, the peak power increases to 20, but TEC decreases to 9,101.01.

Figure 6.1 contrasts the load curves for both solutions and shows a significant difference. For a given minimum makespan, the energy costs vary by 12 % and peak power varies by 25 %. At the same time, the total energy consumption is between 362 and 338. If makespan variation is also considered, one can imagine the amount of possible Pareto-optimal solutions. Therefore, it is beneficial to not identify one good solution but rather

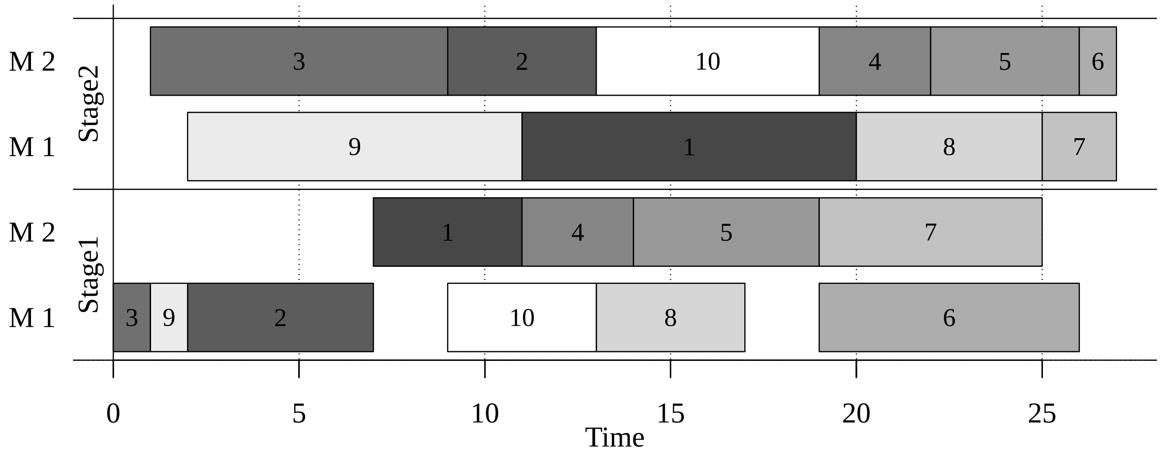


Figure 6.2: Optimal lexicographic solution - lexical order: Makespan ≻ Peak Power ≻ TEC

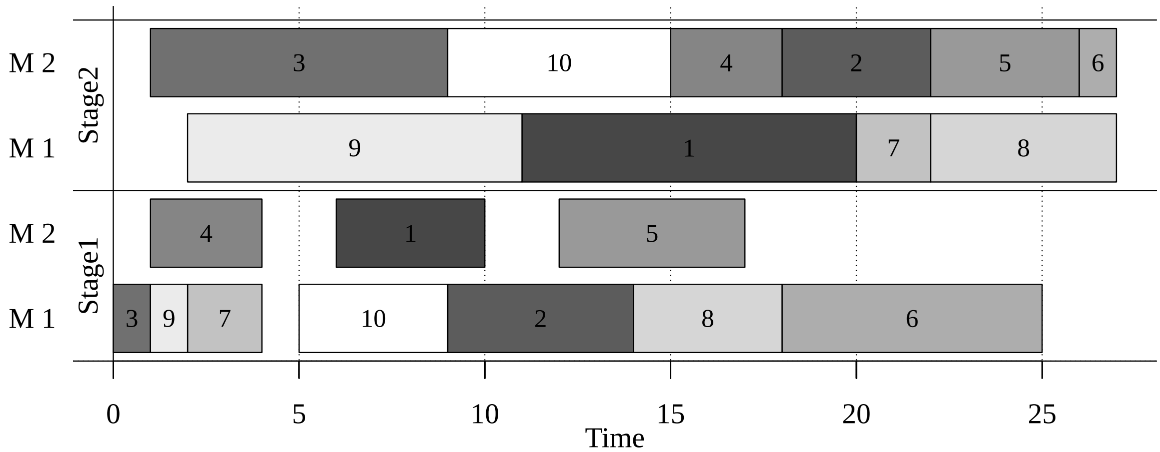


Figure 6.3: Optimal lexicographic solution - lexical order: Makespan ≻ TEC ≻ Peak Power

to point out a set of best possible solutions in order to obtain a solid decision basis. To a certain extent, blending and lexicographic approaches neglect the interdependencies of the different objectives. To avoid this shortcoming, the epsilon method can be used as follows.

Regarding the epsilon method, we first calculate the optimal makespan. With this makespan, a range for peak power is determined whereby the lower bound complies with the minimal peak power for best makespan (similar to the lexicographic solution). The upper bound is defined by the lowest peak power for minimal makespan and the associating minimal TEC (see case 2 in the example above). Then, the best possible TEC is calculated for each peak power value between the lower and upper bounds, resulting in a two-dimensional Pareto front. This is repeated for several given values of makespan, which are gained by a stepwise increase from the optimal makespan, that has been determined in the first step. In doing so, the three-dimensional optimal decision space can be generated. Within this Pareto front, the decision maker can select the best alternative, which depends on the individual case. For this selection, several multi-objective decision-making approaches exist.[33]

It is worth mentioning that for the conventional epsilon method, global minima for TEC and peak power must be calculated, and the makespan must be increased to the corresponding value. Since this value would be very high and is not practically relevant, we restrict the increase of makespan to a lower value.

As Gupta / Hariri / Potts (1997)[34] have shown, the HFSSP is already NP-hard for two production stages with identical parallel machines in the first stage and a single machine in the second stage. Since we consider heterogeneous machines and parallel machines at each stage, the problem considered here is much more complex, can easily be reduced to the problem mentioned above, and is therefore also NP-hard. Obviously, the epsilon method is very time-consuming and not applicable to larger problems; thus, it is not practical for relevant problem sizes. Nevertheless, the epsilon method can be used to evaluate the effectiveness of heuristic algorithms for small-sized instances.

6.4 Iterated local search

Multi-objective optimization aims to find the set of Pareto-optimal solutions. A heuristic algorithm should be capable of improving solutions in-depth (close to Pareto optimality) as well as obtaining diversification to cover the partly very different characteristics of the objectives. For example, both energy costs and peak power can be reduced by decreasing energy consumption, but reducing costs by making use of time-dependent energy prices is in strong contrast to the leveling of energy consumption, which is necessary to keep peak power low. Solutions with low peak power values spread energy-intensive processes over time, while TOU-oriented solutions concentrate tasks at times of low energy costs. Thus, Pareto-optimal solutions can show significant differences. Due to the necessity of diversification, population-based approaches and evolutionary algorithms in particular are preferred to solve multi-objective problems.[35] In contrast, local search approaches concentrate on improving a single solution without maintaining an inherent diverse population. Nevertheless, we will show that an iterated local search approach can be very promising if reference points are used to subdivide the search space.

[33] An overview is given, for example, in Zavadskas / Turskis / Kildiene (2014): *Overviews on MCDM/MADM methods.*
[34] Gupta / Hariri / Potts (1997): *Scheduling a two-stage hybrid flow shop.*
[35] Cf. Ciro et al. (2016): *A NSGA-II and NSGA-III comparison.*

In order to get to different areas of the solution space, the local optimum solution is perturbed. Altogether, the ILS consists of two complementary steps. First, a local search procedure looks for optima in depth. Instead of a simple local search, the described ILS is based on a tabu search in order to avoid reruns. Second, different areas of the solution space are reached by perturbation.[36] Both steps significantly influence the performance of the ILS.

6.4.1 Decoding, encoding & list scheduling

Before a heuristic procedure can be implemented a solution representation must be established. Different types are worth considering. In general, the solution of the considered problem includes two different decisions: on the one hand, the jobs have to be arranged in a certain sequence (permutation); on the other hand, each job must be allocated to a machine at each stage. Complex matrix encoding approaches can be used to display all sequences at all production stages in HFS.[37] However, since this procedure considers a large amount of solutions with a lot of unfavorable schedules, such approaches often lead to poor results.[38] In fact, most algorithms in the literature use a first-stage encoding and schedule further stages through list scheduling algorithms.[39] While the permutation is decoded in the representation of the solution, the machine allocation is determined through the list scheduling procedure. Changes in the sequence in the subsequent production stages can be generated by list scheduling as well.

Based on these thoughts, in order to represent a solution, we use the permutation $\Pi = (\Pi_{(1)}, \Pi_{(2)}, .., \Pi_{(J_{max})})$, with $\Pi_{(j)}$ being the job in the jth position in Π and $j \in J$. Machine allocation and the order in following stages are determined by three different list scheduling approaches. First, the commonly-used earliest completion time (ECT) priority rule is applied.[40] ECT usually results in solutions with good makespan. In order to sufficiently consider energy related objectives, we have developed two new list scheduling algorithms. Both approaches are used for alternative evaluation of the local optima found by ECT. The three algorithms are described in detail as follows.

I Earliest Completion Time

The procedure applied here is inspired by Ruiz / Maroto (2006)[41]. The ECT approach

[36] Cf. Lourenço / Martin / Stützle (2003): *Iterated local search.*
[37] See e.g. Dai et al. (2013): *Energy-efficient scheduling.*
[38] Cf. Ruiz / Maroto (2006): *A genetic algorithm for hybrid flowshops.*
[39] Cf. Shen / Mönch / Buscher (2013): *An iterative approach for the serial batching problem.*
[40] See e.g. Naderi / Ruiz / Zandieh (2010): *Algorithms for a realistic variant of flowshop scheduling.*
[41] Ruiz / Maroto (2006): *A genetic algorithm for hybrid flowshops.*

considers one stage after the other, allocating each job to the machine that leads to the earliest finishing time of the task at the respective stage. To schedule the jobs in the first stage, the finishing time of a job is calculated by the sum of processing times of all preceding jobs on the relevant machine as well as the job's own processing time. In all other stages, the processing time is added to the maximum finishing time of the preceding job on the relevant machine or to the job's own finishing time at the previous stage.

II Deterministic Machine Selection (DMS)

The idea of DMS is to set a fixed allocation for the jobs to one of the parallel machines at each stage, depending on processing time and energy consumption. Therefore, the *standardized machine allocation probability* ($SMAP$) is calculated and then used to assign jobs to certain machines. The basic procedure can be seen in Algorithm 6.1. The product $E_{sj}^m \cdot P_{sj}^m$ corresponds to the total energy demand for a processing step. This value a_{sj}^m is squared to penalize worse machines in terms of energy consumption. Long processing times, and with this a high makespan, are also indirectly penalized, since P_{sj}^m is part of the product. Subsequently, a_{sj}^m is normalized, attaining $SMAP_{sj}^m$, which can be between 0 and 1, where higher values mean better suitability of job j to machine m at stage s.

Algorithm 6.1 List scheduling - deterministic machine selection

procedure CALCULATE MATRIX $SMAP$
  for $s \in S$, $j \in J$, $m \in M_s$ **do**
    $a_{sj}^m = \left(E_{sj}^m \cdot P_{sj}^m\right)^2$
    $SMAP_{sj}^m = \frac{1 - \frac{a_{sj}^m}{\sum_{m' \in M_s} a_{sj}^{m'}}}{M_{s,max} - 1}$
  end for
end procedure
procedure ALLOCATE JOBS TO MACHINES
  for $s \in S$ **do**
    for $m \in M_s$ **do**
      Allocate $\lfloor \frac{J_{max}}{M_{s,max}} \rfloor$ jobs with highest $SMAP_{sj}^m$, which are not
      already allocated at stage s, to machine m.
    end for
    Remaining jobs are processed by the machine with highest $SMAP_{sj}^m$
  end for
end procedure

Based on $SMAP$, the jobs are allocated to different machines at each stage. To avoid all jobs being allocated to the same machine, at first the same number of jobs is assigned to each machine. The number of jobs corresponds to $\lfloor \frac{J_{max}}{M_{s,max}} \rfloor$, thereby considering every machine separately. Then, the remaining jobs are allocated to the most efficient machine, i.e. the machine with the highest value for $SMAP_{sj}^{m}$. This procedure doesn't consider the permutation Π, but since it is deterministic, the assignment only has to be determined once, after which it can be used for all subsequent solutions. DMS shows good results concerning energy costs and peak power; however, because the sequence of jobs is not considered, jobs with high processing times could be assigned to the same machine. This can lead to an unbalanced workload of parallel machines and, therefore, a relatively poor performance regarding makespan.

III Leveling Machine Selection (LMS)

In order to find a list scheduling approach that takes all objectives into account, we introduce a third approach called *Leveling Machine Selection*. In addition to energy efficiency (represented by $SMAP$), LMS also considers machine utilization. The idea is to control job allocation if a machine is already occupied. The finishing times of the last jobs at each stage in the initial solution c_s^{NEH} are used as reference. These values are multiplied by a stretch factor α in order to get an upper bound for the completion time of each job at each stage. The basic procedure can be seen in Algorithm 6.2. The starting schedule c_s^{NEH} is generated by an adapted NEH heuristic, originally introduced by Nawaz / Enscore / Ham (1983)[42] for permutation flow shop scheduling. The exact procedure is explained in detail in Section 6.4.3.

Algorithm 6.2 List scheduling - leveling machine selection

Require: $SMAP_{sj}^{m}$, α, C_s^{NEH}, Π

set $\eta_{sj}^{m} = SMAP_{sj}^{m} \quad \forall\, j, s, m$
for $s \in S$ **do**
    for $j \in \Pi$ **do**
        Generate random number $X \in \left[0; \sum_{m \in M_s} \eta_{sj}^{m}\right]$
        set $a = 0$; set $\mu = 1$
        while $a < X$ **do**
            $a = a + \eta_{sj}^{\mu}$
            $\mu = \mu + 1$
        end while
        Allocate j to machine μ, define C_{js} and remove j from Π
        Update $\eta_{sj}^{\mu} = SMAP_{sj}^{\mu} \cdot \left(1 - \frac{C_{js}}{\alpha \cdot C_s^{NEH}}\right) \quad \forall j \in \Pi$
    end for
    Update Π depending on completion times in s
end for

The LMS works as follows. The probability that a job j is allocated to machine m at

[42] Nawaz / Enscore / Ham (1983): *The m-machine, n-job flow-shop sequencing problem.*

stage s is η^m_{sj}. Initially, that value is equivalent to $SMAP^m_{sj}$. Depending on a random number X, a job is assigned to machine μ. Thereby, X is limited by 0 and the sum of all η^m_{sj} for the considered job in the current stage. Assuming that a job is allocated to any machine ν, the corresponding η^ν_{sj} is subsequently reduced for all jobs that are not already scheduled on the basis of utilization of machine ν. Thus, the probabilities decrease with the number of considered jobs. The aim of that procedure is to distribute the jobs not only by energy consumption but also to ensure even assignment to parallel machines to obtain good makespan results.

With this approach, solutions are generated that take all three objectives into account. Since this procedure integrates stochastic elements, it is beneficial to generate various LMS solutions for the same permutation Π. The exact number of repetitions is a parameter of the algorithm.

6.4.2 Right shifting improvement

All three list scheduling algorithms allocate the jobs to one of the parallel machines at each stage. The processing begins as soon as the job is ready and the previous jobs at the stage are finished. Nevertheless, it is possible to make use of idle times to shift jobs and to thus reduce time-dependent energy costs. Simultaneously, job postponements may decrease peak power. To exploit this potential, a right shifting algorithm is used, which examines possible cost reductions.[43]

Algorithm 6.3 Right shifting improvement procedure

Require: π^m_s - sequence of jobs on machine m at stage s
for $s \in \{S_{max}, ..., 1\},\ m \in M_s,\ j \in reverse(\pi^m_s)$ **do**
    $slack :=$ possible postponement of j
    set $a = 0$; set $b = 0$; set $\mu = 0$
    for $\tau \in \{1, ..., slack\}$ **do**
        $a := a + RTP_{c_{sj}+\tau} - RTP_{c_{sj}+\tau-P^m_{sj}-1}$
        if $a < b$ **then**
            $b := a;\ \mu := \tau$
        end if
    end for
    Shift j for μ time units
end for

The general procedure is displayed in Algorithm 6.3. Essentially, we test all tasks that would not lead to an increase of makespan by starting later and postponing ensuing tasks.

[43]A similar approach can be found (for example) in Luo et al. (2013): *Hybrid flow shop scheduling.*

The tasks not to be considered describe the critical path of the representation of the problem as a disjunctive graph. Beginning at the last stage, machines are considered one after the other. For each machine, the job sequence π_s^m is known and handled in reverse order. If a job can be processed later without influencing a following job, the next work step or the makespan, the maximum shift is defined as *slack*. All possible moves are tested, and the most cost-efficient production time for j is determined. To minimize computational effort, we do not calculate TEC each time, but only differences in RTP_t. If a job is fixed, the previous one is considered in the same way.

Theoretically, it could be beneficial to shift various jobs at the same time. For example, if postponing of a job increases TEC slightly, the job will not be changed, even if the same postponement of the previous job could reduce TEC significantly; as a result, the variation is beneficial. However, verification of all neighboring job combinations would greatly increase the computational effort. For this reason, we do not consider such interdependencies here.

6.4.3 Initial solution

The idea of our reference-point based ILS is to start with a good makespan solution and then to stepwise guide the search to energy related objectives. For the makespan criterion in HFS, a lot of different constructive algorithms have been published. The NEH heuristic was originally developed for the permutation flow shop, but since Brah / Loo (1999)[44] first adopted it to HFS, it has become known as one of the best and most efficient approaches. To generate an initial solution, NEH is applied as it is in Jungwattanakit et al. (2008)[45]. The total processing time P_j of each job is defined by the sum of processing times over all stages and parallel machines:

$$P_j = \sum_{s \in S} \sum_{m \in M_s} P_{sj}^m \tag{6.20}$$

If the number of parallel machines varies widely at different production stages, it could be useful to sum up the average processing times at each stage. Since the number of parallel machines at each stage is equal in our test instances, we do not apply this adjustment here.

The application of NEH to HFS according to Jungwattanakit et al. (2008) considers all permutations of the two jobs with highest total processing times. Afterwards, in decreasing order of total processing time, jobs are inserted at each possible position in the permutation and set to the best one. We make a small adaptation by considering

[44] Brah / Loo (1999): *Heuristics for scheduling.*
[45] Jungwattanakit et al. (2008): *Algorithms for flexible flow shop problems.*

all possible permutations of the first four jobs, which leads to an improvement of the initial solution. As described in Section 6.4.1, NEH results are not only used for the ILS procedure itself, but also as a reference for the LMS list scheduling. The completion time of the last job at each stage in the NEH solution is saved as c_s^{NEH}.

6.4.4 General ILS procedure

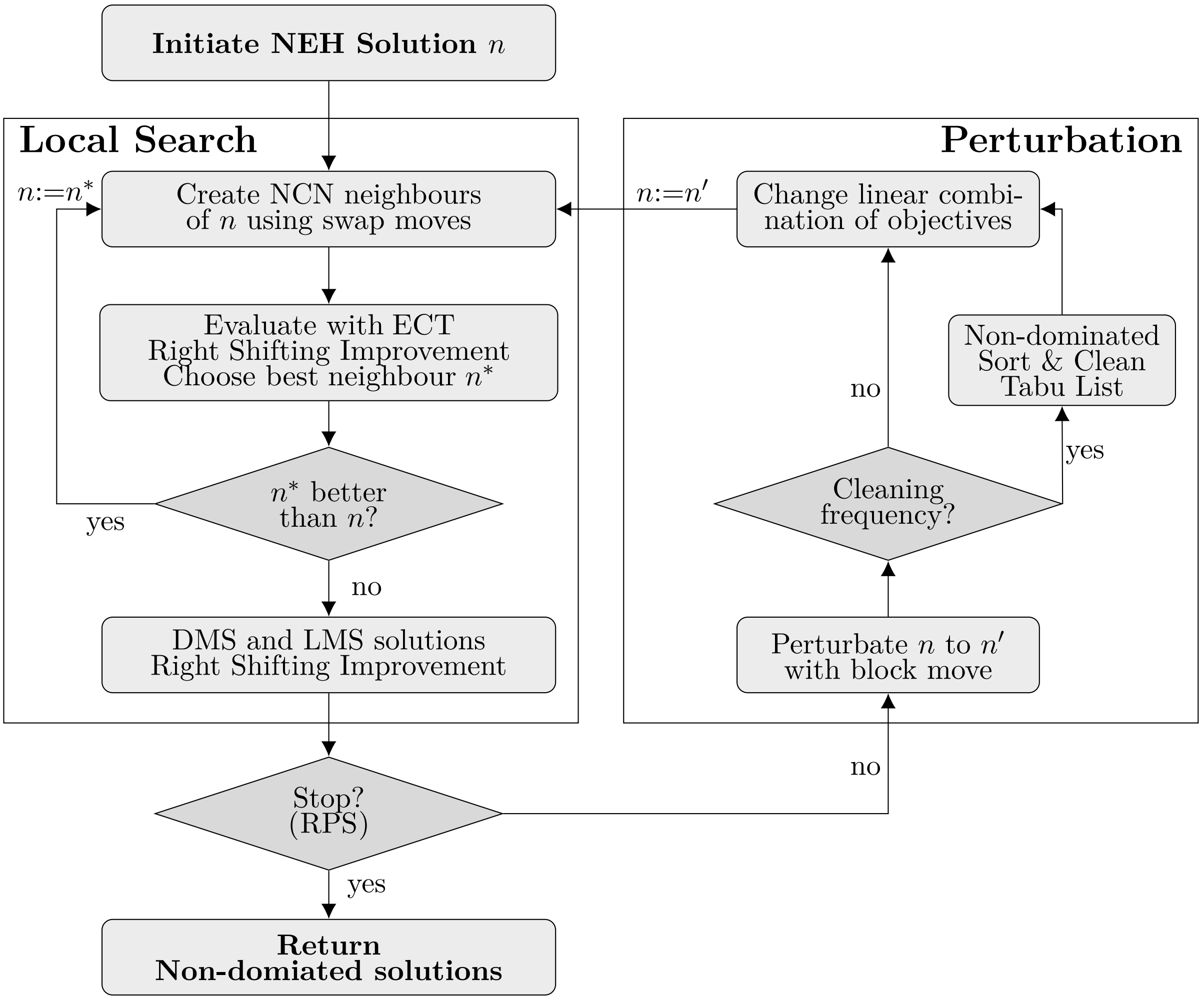


Figure 6.4: ILS procedure - flow diagram

The general procedure of the ILS is illustrated in Figure 6.4. After generating an initial solution, we use classical swap moves for the following local search, which means that we change the position of two randomly-chosen jobs in the permutation. Furthermore, we integrate a tabu list to avoid repetitions. Altogether, the neighborhood consists of $\frac{(J_{max})^2 - J_{max}}{2}$ possible solutions. For small instances, it is possible to take the complete neighborhood into account, but for 20 or more jobs, we limit the number of observed swaps to reduce the computational effort. The number of considered neighbors (NCN) will be used as a parameter to control the heuristic.

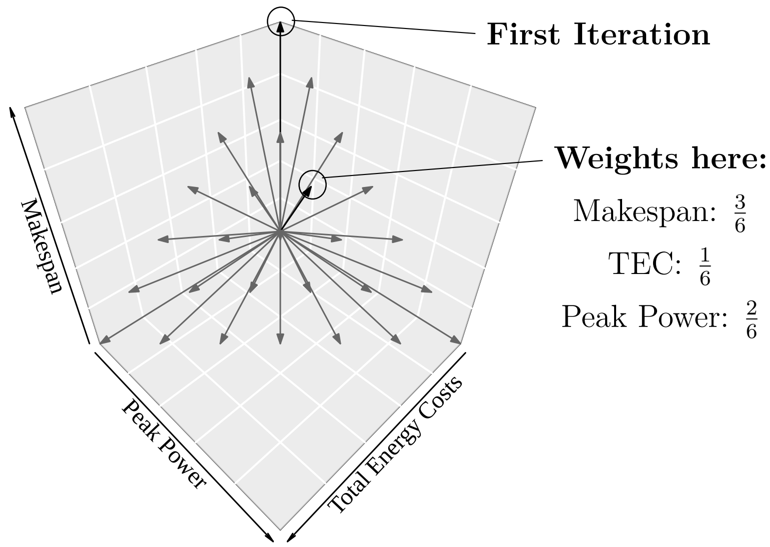


Figure 6.5: Local search directions for RPS=6

For each considered neighbor, the objectives are calculated using ECT. Then, each solution is evaluated by a fitness function, and the best permutation is selected. Due to the multiple objectives, the acceptance criterion is a critical feature of the local search procedure. Here, reference points are used to weight the different objectives. To consider all different areas of the solution space the reference points are distributed evenly. An example is shown in Figure 6.5. The number of nuances between two objectives is called reference point sharpness (RPS) and determines the level of detail. The total number of local search runs can be calculated by $\frac{1}{2} \cdot RPS^2 + 1.5 \cdot RPS + 1$.

In the first iteration, only makespan is chosen (see Figure 6.5). Afterwards, energy-related objectives gain increasing importance by using different linear combinations of the three objectives. In Figure 6.5, one reference point is highlighted with weighting factors. The associated fitness function is given in Equation 6.21. Since the objectives' magnitudes are quite different, it is necessary to normalize the values. This is achieved by setting the neighbors' objectives n_i in relation to the incumbent solution of the local search n. For example, $C_{max}(n_i)$ is the makespan of a neighbor $i \in \{1, ..., NCN\}$ and divided by the current value $C_{max}(n)$.

$$Fitness = \frac{3}{6} \cdot \frac{C_{max}(n_i)}{C_{max}(n)} + \frac{1}{6} \cdot \frac{TEC(n_i)}{TEC(n)} + \frac{2}{6} \cdot \frac{PP(n_i)}{PP(n)} \tag{6.21}$$

The neighbor with best fitness n^* is used for the next iteration. If no neighbor is better than the incumbent permutation, the local search is left, and DMS and various LMS solutions are calculated. Afterwards, the stopping criterion is checked. If not all reference points were used, the current solution is perturbed.

In Figure 6.4, the perturbation is shown on the right side. Perturbation is used to obtain diversification and to reach different areas of the solution space. At the same time, the general quality of the incumbent solution should be preserved. Hence, a simple restart with a random initial solution may not be promising. Xu / Weng / Fujimura (2014)[46] demonstrate that a block move neighborhood leads to very good results for sequence decoding; therefore, we will use a block move procedure to disrupt the incumbent solution. The length of a block is randomly chosen in the range of 2 and an upper bound parameter.

As already mentioned, all solutions are saved to a tabu list to avoid repeated generations of the same solution. However, a solution that was rejected for a given linear combination of objectives could be beneficial in other search directions. Therefore, we have to regularly clear the list. To do this, after a defined number of iterations, a non-dominated sort is executed. All dominated solutions are deleted, and only non-dominated solutions are kept. After choosing a new reference point, the local search restarts with the perturbed solution n', and the cyclic process proceeds.

6.5 Experimental setting

6.5.1 Algorithms and parameter setting

To analyze the performance of the proposed heuristic, ILS results are compared to the optimal Pareto front calculated with the presented MIP in Section 6.3.1. Of course, only small problem instances can be solved to optimality. Since the calculation of a single optimal schedule takes several days, this procedure is only suitable to roughly estimate the solution quality. For medium and large problem sizes, we use the widely-known NSGA2 as a benchmark.[47] The crossover is done by PMX (partial-mapped-crossover). For mutation, swap moves are used.

The performance of the heuristics largely depends on the chosen parameters. An overview of the used values is given in Table 6.2. To determine the parameters for ILS, random initial combinations of parameters have been tested. The results show that primarily RPS and NCN influence the performance. Due to lower importance, other values are set after the first test runs. In detail, the tabu list is cleared after 5 block moves in order to be able to accept already-rejected solutions for other objective combinations. Each local optimum

[46] Xu / Weng / Fujimura (2014): *Energy-Efficient Scheduling.*
[47] See e.g. Deb et al. (2002): *NSGA-II.*

is scheduled ten times with the LMS list scheduling approach. The length of a block during the perturbation is between two and the square root of the number of jobs.

ILS		NSGA2	
Reference Point Sharpness	35	Crossover probability	1.0
Considered neighbors (NCN)	100	Mutation probability	0.2
Solution cleaning frequency	5	Number of generations	1000
LMS reruns	10	Population Size	500*
Maximum block size	$\sqrt{J_{max}}$		
*For 10 job instances a population size of 200 was used			

Table 6.2: Used parameters for the heuristics

To find appropriate values for RPS and NCN, an extensive study has been conducted (for detailed results see supplementary data). RPS is varied from 15 to 50 at intervals of 5, and NCN is set to 50, 100, 150, 200, 250, and 300. Thus, 48 different parameter settings are considered. These combinations are used for 18 different instances with 20 and 50 jobs, as well as 2, 3, and 4 stages and respectively 2, 3, and 4 parallel machines. Each instance is solved 10 times to exclude random effects. An RPS of 35 and an NCN of 100 lead to promising results regarding calculation time and solution quality. The NSGA2 parameters are inspired by the settings of Luo et al. (2013)[48]. Moreover, they are chosen so that the required computation time is similar to ILS. This allows for a good comparison of the algorithms.

6.5.2 Performance criteria

There are many different quality indicators for multi-objective optimization problems. To evaluate the performance of the proposed algorithm we use two different criteria:

1. **Number of non-dominated solutions (NDS)**
   Compared to a solution vector $\bar{y}$, a solution $\bar{x}$ that is worse regarding at least one objective and not better regarding any other objective is considered to be dominated by $\bar{y}$. Vice versa, $\bar{x}$ is non-dominated if every other solution is worse in at least one objective. Each non-dominated solution is consequently Pareto-optimal, and all non-dominated solutions together depict the best-known decision possibilities. An example of dominated and non-dominated solutions can be seen in Figure 6.6. The picture is depicted for a bi-objective problem to facilitate understanding, but it would

[48] Luo et al. (2013): *Hybrid flow shop scheduling.*

be similar for the three-objective case. NDS measures the solution quantity but, unfortunately, does not provide any information about solution quality or deviation from optimum. Therefore, hypervolume is considered as a second criterion.

2. **Hypervolume (HV)**
Hypervolume is one of the most popular multi-objective performance criteria, and holds outstanding importance.[49] It was introduced by Zitzler / Thiele (1998)[50] and originally called *size of dominated space*. The basic premise is to estimate the normalized amount of space that is dominated by Pareto-optimal solutions; therefore, the possible solution space, bounded by the theoretical optimal and anti-optimal points for each objective, is calculated. Both points can be seen in Figure 6.6. The theoretical optimal point contains all best-known single objective values (in graphic intersection of LB1 and LB2), while the anti-optimal point consists of the worst objective values over all non-dominated solutions (intersection of UB1 and UB2). The part of that space that is covered by non-dominated solutions is called hypervolume. This area is colored gray in Figure 6.6 and divides the solution space, which is visualized by dashed lines. The hypervolume can range between 0 and 1, whereby 1 is the theoretical best possible value. To accurately estimate the solution space, all algorithms are run several times for each test instance, considering all results to define the optimal and anti-optimal points.

6.5.3 Test instances

To the best of our knowledge, no benchmark instances are available for the considered problem. For this reason, we generate own instances. Since Luo et al. (2013)[51] consider a similar problem, most of our values are based on their work. Altogether, 45 different problem sizes are considered. A summary is given in Table 6.3. For each problem size, we generate 30 different random instances. Each instance is solved 10 times with both algorithms, in order to exclude random outliers in hypervolume calculation. Thus, 13,500 runs in total are examined for ILS and NSGA2. For energy costs, we use Phelix spot market prices from 27-31 March, 2017. All tests are run on an Intel Xeon 3.3 GHz CPU with 768 GB memory. While CPLEX 12.6 is used for the optimal Pareto front in Section 6.6.1, the algorithms ILS and NSGA2 are coded in C# programming language with Microsoft Visual Studio 2013.

[49] Cf. Beume / Naujoks / Emmerich (2007): *Dominated hypervolume.*
[50] Zitzler / Thiele (1998): *Multiobjective Optimization.*
[51] Luo et al. (2013): *Hybrid flow shop scheduling.*

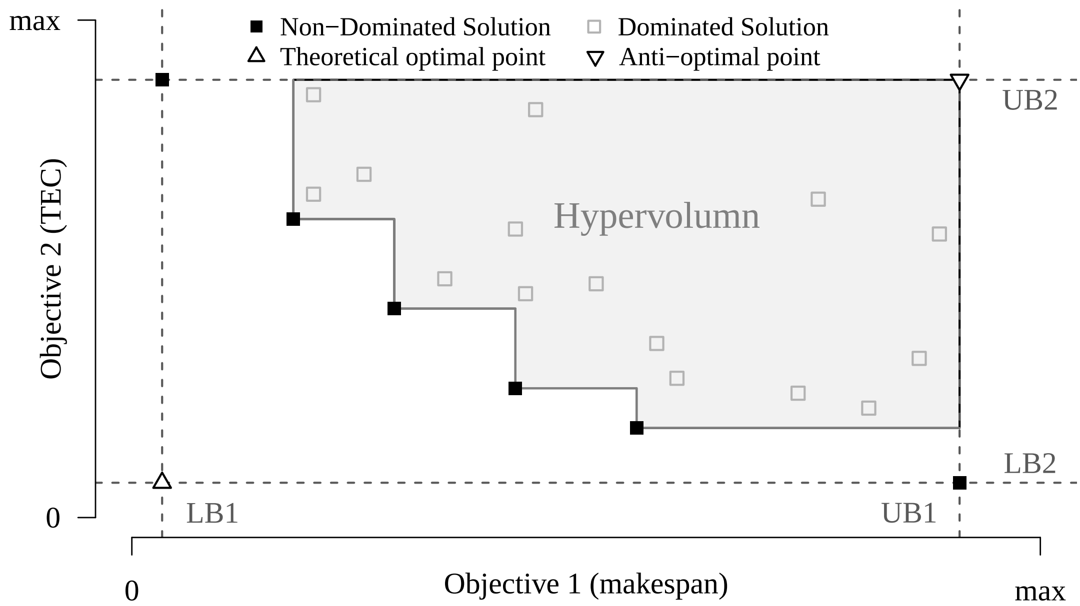


Figure 6.6: Hypervolume and non-dominated solutions for a bi-objective problem

Parameter	Levels
Number of jobs	10, 20, 30, 50, 100
Number of stages	2, 3, 4
Number of parallel machines	2, 3, 4
Processing time P_{sj}^m	$unif\{1; 10\}$
Energy demand E_{sj}^m	$unif\{1; 10\}$

Table 6.3: Summary of test instances

6.6 Computational results

6.6.1 Optimal solution with the epsilon method

To analyze the problem structure and the interdependencies of the different objectives, we consider the instance that has already been introduced in Section 6.3.2, with 10 jobs on 2 production stages with 2 parallel machines at each stage. This problem is solved using ILS and NSGA2 (10 runs each) as well as the epsilon method described in Section 6.3.2. The results can be seen in Table 6.4.

Computation times of the heuristic approaches are much smaller. Since the generation of the optimal Pareto front takes several days for this problem, it is not relevant for practical problem sizes. However, the results for NDS and hypervolume differ significantly. A detailed illustration of the optimal pareto front and the results of a single ILS run

	CPU Time			Non-Dom. Sol.			Hypervolume		
	min	mean	max	min	mean	max	min	mean	max
Optimal	-	67h	-	-	209	-	-	0.785	-
ILS	1.21s	1.26s	1.31s	34	39.2	48	0.604	0.631	0.647
NSGA2	3.03s	3.14s	3.25s	8	9.9	12	0.461	0.462	0.466

Table 6.4: Comparison of optimal and ILS solution

can be seen in Figure 6.7. (Please note that a single grid point represents the minimum energy costs found for a combination of peak power and makespan, and therefore, not all grid points represent NDS.) The optimal Pareto front reaches lower energy costs in particular. While the ILS nearly finds the optimal makespan of 27 (ILS = 28) and detects the minimum peak power, on average, the ILS energy costs are at around 10.3 % above the optimal solution. One reason could be the encoding procedure: because only the first-stage permutation is encoded, the overtaking of jobs in following stages is probably neglected, even though it could lead to better solutions. In addition, the right shifting could be improved. Since the shifting of several jobs together is not considered, some improvement potential cannot be realized here. Furthermore, Figure 6.7 shows that ILS does not examine the area of high makespan well. Due to the time oriented list scheduling approach, these areas of the solution space are, likewise, not considered. For practical problems, it is questionable whether these solutions are highly relevant. Accordingly, the energy costs differ just 7.3 % for makespan ≤ 40.

For very restrictive PP values, the ILS solution also seems to deviate more strongly from the optimal costs. In order to consider this area more closely, the right shifting approach could be extended or an additional levelling approach could be integrated. Since in this area energy costs or makespan must inevitably rise significantly and the computing effort would increase enormously, this is not done at this point. Nevertheless, these approaches could be discussed in a future work. Overall, it must be kept in mind that here only a single instance shows that the ILS comes relatively close to the optimal values in the interesting range with low makespan. For statistical significance, however, several months of computational study would be necessary and only statements for very small instances would be possible, while no information exists about the scalability.

6.6.2 Comparison of ILS and NSGA2

All test instances described in Section 6.5.3 were solved with the proposed ILS and with NSGA2. The results are summarized in Table 6.5, organized by problem size. Since 30

Instance*			**ILS**							**NSGA2**						
J	**S**	**M**	t_{CPU}	**NDS**			**HV**			t_{CPU}	**NDS**			**HV**		
				Min	Mean	Max	Min	Mean	Max		Min	Mean	Max	Min	Mean	Max
10	2	2	**1.2**	**28.3**	**38.6**	**47.4**	**0.72**	**0.77**	**0.79**	3.0	8.3	10.1	13.2	0.56	0.56	0.57
10	2	3	**1.5**	21.4	**38.7**	**45.4**	**0.70**	**0.73**	**0.76**	3.2	**34.1**	36.5	38.7	0.70	0.70	0.70
10	2	4	**1.7**	**23.0**	**25.6**	**28.6**	**0.74**	**0.76**	**0.78**	3.2	20.2	21.5	22.6	0.70	0.72	0.74
10	3	2	**1.6**	**36.0**	**42.8**	**50.8**	**0.71**	**0.72**	**0.74**	3.3	25.1	29.8	35.2	0.62	0.65	0.67
10	3	3	**1.9**	**44.0**	**50.6**	**57.7**	**0.78**	**0.79**	**0.80**	3.6	37.4	43.1	49.6	0.74	0.76	0.78
10	3	4	**2.1**	29.6	34.0	39.0	**0.80**	**0.82**	**0.83**	3.7	**31.7**	**35.8**	**39.9**	0.78	0.79	0.81
10	4	2	**1.9**	**62.0**	**74.0**	**85.7**	**0.72**	**0.73**	**0.75**	3.8	34.4	40.9	46.7	0.59	0.61	0.62
10	4	3	**2.2**	**37.3**	**45.4**	**53.6**	**0.73**	**0.75**	**0.76**	3.9	16.1	18.4	20.6	0.50	0.51	0.51
10	4	4	**2.6**	**24.9**	**40.0**	**47.4**	**0.70**	**0.72**	**0.75**	4.2	22.4	26.4	30.4	0.57	0.58	0.61
20	2	2	**6.5**	54.7	65.4	76.2	0.69	0.72	0.74	17.4	**54.8**	**68.3**	**83.6**	**0.70**	**0.73**	**0.76**
20	2	3	**7.1**	**24.2**	**31.5**	**39.7**	**0.71**	**0.74**	0.76	17.7	18.4	26.7	36.5	0.69	0.73	**0.77**
20	2	4	**9.6**	23.1	30.6	39.2	0.73	0.77	0.80	18.2	**25.1**	**33.9**	**43.0**	**0.74**	**0.79**	**0.84**
20	3	2	**7.8**	63.2	77.6	92.5	**0.72**	**0.74**	0.76	18.8	**69.2**	**89.6**	**112.6**	0.69	0.73	**0.78**
20	3	3	**7.9**	**37.0**	**47.7**	**58.8**	**0.67**	**0.70**	**0.72**	19.7	17.9	28.1	41.0	0.64	0.67	0.70
20	3	4	**9.4**	**38.8**	**50.3**	**61.1**	**0.62**	0.65	0.68	20.9	18.2	28.3	38.9	0.62	**0.66**	**0.70**
20	4	2	**8.1**	44.8	59.5	73.1	0.67	0.70	0.72	21.3	**47.1**	**68.8**	**94.0**	**0.68**	**0.72**	**0.77**
20	4	3	**9.4**	**55.7**	**71.2**	**86.5**	**0.72**	**0.74**	**0.75**	22.1	25.6	42.3	58.9	0.57	0.61	0.64
20	4	4	**11.0**	**35.9**	**46.6**	**57.9**	**0.64**	0.68	0.72	22.3	19.6	31.6	45.1	0.64	**0.68**	**0.72**
30	2	2	**11.8**	35.1	45.7	56.5	**0.71**	0.74	0.77	19.7	**45.6**	**57.6**	**70.8**	0.70	**0.75**	**0.80**
30	2	3	**13.3**	26.3	37.1	49.5	**0.72**	0.76	0.79	19.9	**34.4**	**49.3**	**66.5**	0.71	**0.77**	**0.84**
30	2	4	**15.8**	**38.1**	**53.6**	**67.8**	**0.70**	0.73	0.75	20.6	17.3	29.0	40.9	0.67	**0.74**	**0.81**
30	3	2	**12.3**	**56.6**	**69.4**	**84.1**	**0.69**	**0.71**	0.74	22.7	33.3	48.0	68.2	0.63	0.70	**0.76**
30	3	3	**14.1**	**52.0**	**65.1**	**81.5**	**0.68**	0.70	0.73	22.9	38.8	57.7	80.2	0.65	**0.71**	**0.78**
30	3	4	**15.9**	**38.3**	**50.4**	**62.9**	**0.59**	**0.62**	0.66	23.8	18.1	30.4	44.8	0.53	0.61	**0.68**
30	4	2	**13.8**	**51.1**	**66.7**	**82.9**	**0.69**	0.71	0.74	24.8	30.1	45.6	66.0	0.67	**0.72**	**0.77**
30	4	3	**16.2**	**54.6**	**71.7**	89.3	**0.66**	**0.68**	0.71	25.8	46.7	70.5	**94.9**	0.59	0.65	**0.72**
30	4	4	**18.0**	**49.1**	**64.9**	**81.8**	**0.62**	**0.64**	**0.67**	26.2	29.1	49.4	71.0	0.48	0.52	0.57
50	2	2	**20.8**	**37.2**	**49.4**	61.7	**0.75**	0.78	0.81	23.8	32.5	46.1	61.7	0.75	**0.81**	**0.88**
50	2	3	**20.9**	**34.6**	**46.8**	58.9	**0.74**	0.77	0.79	24.4	25.1	42.7	**64.1**	0.69	**0.77**	**0.84**
50	2	4	**24.2**	**40.9**	**52.1**	**64.4**	**0.66**	**0.68**	**0.70**	26.2	29.7	44.3	60.4	0.52	0.59	0.66
50	3	2	**20.5**	**55.5**	**69.6**	**84.7**	0.66	0.69	0.72	28.4	42.9	59.2	77.3	**0.69**	**0.74**	**0.80**
50	3	3	**23.0**	**57.7**	**73.4**	**89.9**	**0.67**	**0.70**	0.72	29.6	32.0	47.0	66.8	0.60	0.66	**0.73**
50	3	4	**27.5**	**52.3**	**68.1**	**82.2**	**0.63**	**0.66**	**0.68**	30.6	37.6	57.5	81.0	0.43	0.50	0.57
50	4	2	**23.5**	**58.6**	**76.2**	**94.3**	**0.68**	**0.70**	0.73	32.4	31.2	47.8	67.3	0.63	0.69	**0.77**
50	4	3	**26.7**	**53.0**	**69.7**	**87.3**	**0.69**	**0.71**	**0.74**	34.5	24.6	39.7	55.8	0.60	0.66	0.72
50	4	4	**29.5**	**46.1**	**62.8**	**79.8**	**0.56**	**0.59**	**0.63**	35.3	27.8	49.8	73.6	0.42	0.49	0.56
100	2	2	37.2	**51.9**	**68.3**	**85.4**	**0.82**	**0.85**	**0.88**	**33.4**	24.5	39.3	57.4	0.59	0.65	0.71
100	2	3	**37.1**	**31.3**	**43.1**	**57.3**	0.78	0.81	0.83	37.3	16.3	28.1	43.3	**0.79**	**0.85**	**0.91**
100	2	4	43.6	**25.4**	**37.1**	**49.2**	**0.80**	0.82	0.86	**37.8**	18.9	31.2	46.2	0.79	**0.84**	**0.89**
100	3	2	**41.8**	**68.2**	**88.6**	**109.7**	**0.63**	**0.66**	**0.69**	42.1	28.7	49.5	73.9	0.53	0.58	0.63
100	3	3	**41.6**	**46.1**	**60.1**	**78.4**	**0.83**	**0.85**	0.87	46.7	19.8	32.2	47.7	0.79	0.83	**0.88**
100	3	4	**44.5**	**39.2**	**53.4**	**71.1**	0.70	0.73	0.76	50.3	15.2	30.3	49.0	**0.71**	**0.76**	**0.82**
100	4	2	**47.2**	**76.9**	**95.8**	**118.3**	**0.67**	**0.69**	**0.72**	48.7	33.8	60.4	91.0	0.44	0.49	0.54
100	4	3	**45.9**	**49.0**	**68.5**	**90.2**	**0.78**	**0.81**	**0.84**	56.9	14.7	29.7	44.8	0.72	0.77	0.81
100	4	4	**49.4**	**46.4**	**62.7**	**81.2**	**0.79**	**0.82**	**0.84**	58.9	18.4	36.4	58.2	0.71	0.75	0.80
Total			**18.4**	**43.4**	**56.5**	**69.8**	**0.70**	**0.73**	**0.76**	**24.3**	**28.7**	**42.0**	**57.2**	**0.64**	**0.69**	**0.73**

*Abbreviations: J=number of jobs, S=number of stages, M= number of parallel machines; t_{CPU} in sec.

Table 6.5: Computational results

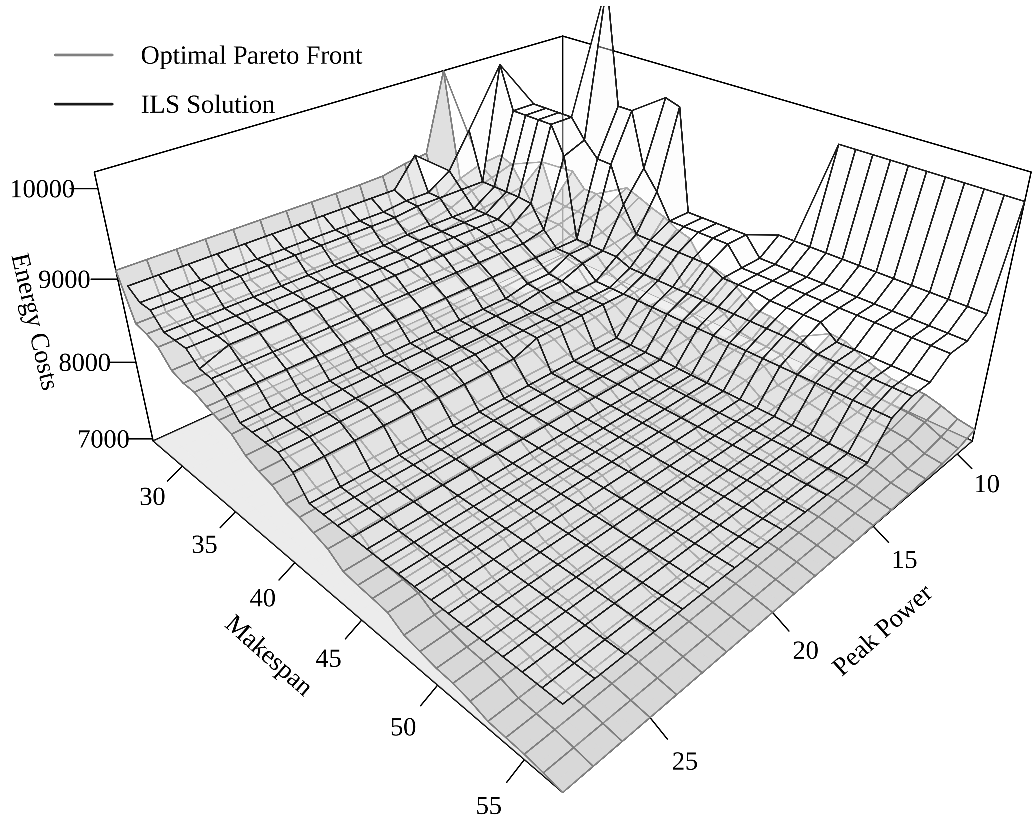


Figure 6.7: Optimal pareto front vs. ILS solution

instances were solved 10 times for each problem size, each line represents 300 runs. To consider the spread of solution quality, the extrema for NDS and HV are also given. The results are analyzed regarding both comparison criteria.

Non-dominated solutions

Because of higher computation times, the NSGA2 finds more NDS in some mid-size problems, but overall the ILS is capable of identifying more diversified solutions. While the NSGA2 determines 42 NDS on average, the ILS generates more than 56. Apparently, the progressive change of the fitness function through reference points allows for a more effective propagation of the solution space. As shown in Figure 6.8, the number of non-dominated solutions generally increases by the number of stages as well as the number of jobs, and the solution space expands as well. At the same time, a higher number of parallel machines leads to less NDS. Figure 6.8 also displays that, for most problem sizes, the average number of NDS is similar to the value gained by the worst run of ILS (the dashed lines indicate the range with minimum and maximum values). This proves the effectiveness of ILS regarding NDS.

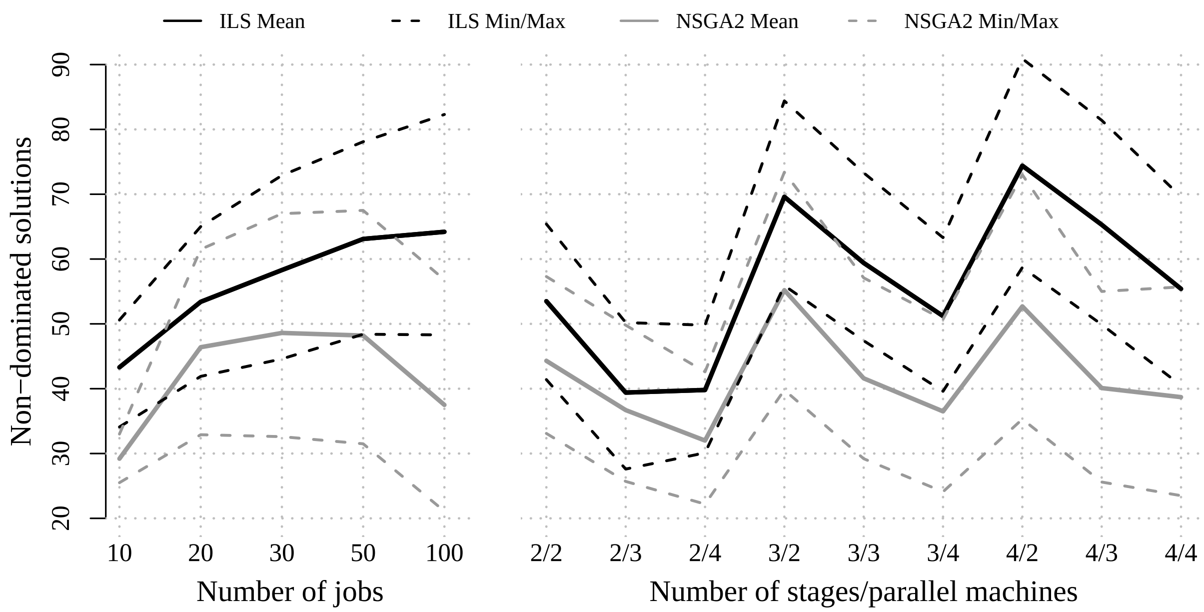


Figure 6.8: Non-dominated solutions depending on problem size

Hypervolume

The differences in HV are not as clear as for NDS; nevertheless, ILS significantly outperforms NSGA2. In total, the ILS determines better solutions by 4 % (total average of 0.73 compared to 0.69) and is superior in 29 of the 45 instances. The resulting average HV (subject to the problem size) is illustrated in Figure 6.9. It can be seen that, with one exception, the ILS solutions obtain better HV. The spread of the NSGA2 is clearly wider (cf. dashed lines), which means that it is subject to more random influences and would probably need more computation time to more closely examine. The maximum values of NSGA2 are above the ILS maximum values for 2 stages but drop below the minimum value of ILS for 4 stages. A similar behavior is observed for the number of jobs. It can be deduced that the ILS is particularly favorable for bigger problem sizes. This contradicts the general idea that evolutionary algorithms are particularly beneficial to large solution spaces, where it is worthwhile to generate and modify a population.

Computation time

With an average of less than 60 seconds for all problem sizes, the computational effort remains reasonable for both metaheuristics. NSGA2 works considerably slower than the proposed ILS algorithm, requiring approximately one-third additional CPU time (total average 24.3 s, compared to 18.4 s). Only 2 of the 45 problem sizes can be solved faster by the NSGA2 compared to ILS. Particularly for mid-size problems, the NSGA2 requires

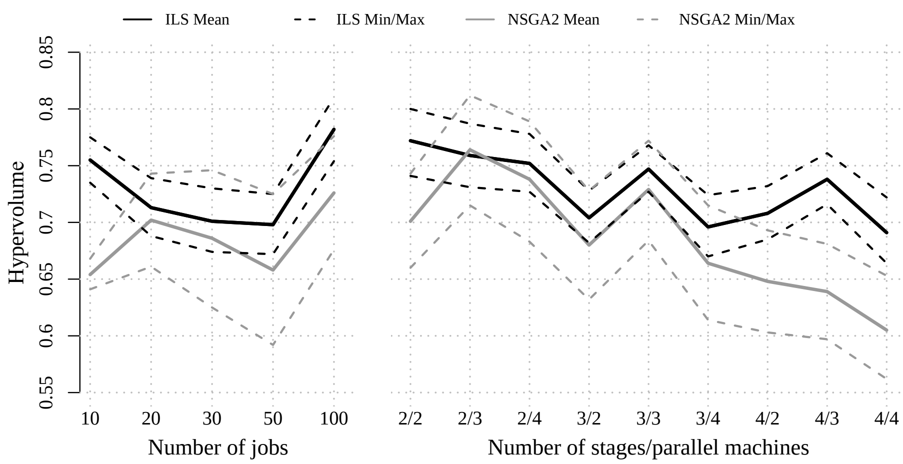


Figure 6.9: Hypervolume depending on problem size

significantly higher computation times. As mentioned above, these problem instances are those where the solution qualities of ILS and NSGA2 converge. This shows that further improvement of solution quality is possible at the cost of higher computational effort if the algorithms' parameters are adjusted. Understandably, CPU time rises with problem size.

6.6.3 Statistical evaluation

In computational studies, it is possible for differences in performance criteria to occur simply by chance. Due to the high number of computational runs in this study, the probability is very low. Nevertheless, one should test whether the advantages of the ILS are significant and whether the ILS performs better than the NSGA2 on average. For this purpose, a dependent t-test for paired samples has been conducted. The null hypothesis H_0 indicates that the mean performances of both algorithms are the same. Due to the high sample size, the significance level is set at one percent. The test results are displayed in Table 6.6.

H_0	t-value	df	p-value	result
NDS_ILS = NDS_NSGA2	6.9452	44	$1.4 \cdot 10^{-8}$	NDS_ILS > NDS_NSGA2
HV_ILS = HV_NSGA2	4.1619	44	$1.4 \cdot 10^{-4}$	HV_ILS > HV_NSGA2
CPU_ILS = CPU_NSGA2	-8.5897	44	$5.8 \cdot 10^{-11}$	CPU_ILS < CPU_NSGA2

Table 6.6: Dependent t-test for paired samples of mean values

The mean values of Table 6.5 are used for calculation. The p-value describes the significance level needed to reject the null hypothesis. Since this value is always considerably below one percent, all statements made can be decisively confirmed by the test results. It can be noted that the ILS will find more non-dominated solutions on average in a shorter period of time and covering a larger area of the solution space (hypervolume).

6.7 Summary

Three strategies are particularly important for energy-aware scheduling. EAS is a matter of reducing energy consumption, exploiting varying energy prices to reduce energy costs, and leveling energy consumption in order to avoid expensive energy peaks. To the best of our knowledge, this is the first time that all three strategies are addressed in one approach. Specifically, an energy-aware hybrid flow shop scheduling problem has been presented that aims to simultaneously minimize makespan, total energy costs, and peak load. The underlying problem is modeled as a mixed integer problem (MIP) in order to solve small instances to optimality. Due to the multi-criteria objective function a lexicographic approach is used to illustrate the influence of different priority orders on the optimal schedule. Then, the epsilon method serves to identify the optimal Pareto front for small problems and to compare them with the solutions of the newly-developed heuristic. To verify the performance of our new ILS for larger instances the results are compared to solutions determined with NSGA2, which represents the state-of-the-art.

Through numerical examples and a large computational study with 13,500 solved problems, we achieve the following main findings: (i) As expected, a lower makespan leads to higher total energy costs for a given peak load. A short makespan forces jobs to be scheduled even if energy prices are comparatively high. (ii) Reducing the peak load will initially lead to a slight increase in costs. Only above a certain limit does a strong restrictive effect appear, so that the heuristics can only determine feasible solutions with high costs without increasing lead time. (iii) Not surprisingly, the costs are highest when both, makespan and peak load, are low. (iv) Methodologically, we have shown that the newly-developed ILS is superior to NSGA2 in terms of both, non-dominated solutions and hypervolume. It is noteworthy that, for this multi-criteria energy-aware scheduling problem, a method based on local search leads to better results than the conventional NSGA2 method, which was developed especially for this purpose. However, this required a corresponding adaptation of the ILS presented here.

Methodological improvements of the presented ILS could be made in future work. We have already mentioned that the right shifting procedure could be carried out not only for one job but for several at the same time. In addition, the search for Pareto-optimal solutions in the area of higher makespans could be intensified, despite the question of practical relevance. Furthermore, the parameter setting could be improved by adapting it to the problem size.

Depending on the problem at hand, other aspects relevant to planning may need to be considered. For example, it might be advantageous to explicitly consider the cost effects of switching machines on and off. It may also be possible to speed up the processing of individual jobs. However, the advantage of shorter processing times is often offset by the disadvantage of higher energy consumption.

Last but not least, in this study we have assumed that the fluctuating energy prices are known. However, the scheduling of jobs requires an energy price forecast that is as accurate as possible. It would undoubtedly be desirable to embed such a forecast in an integrated approach.

7 Hybrid flow Shop scheduling with subcontracting options and time-depending energy costs

Abstract

Motivated by the increasing requirement for flexibility in industrial manufacturing this paper analyzes the influence of subcontracting options in production scheduling. Therefore, a comprehensive MIP formulation for a general hybrid flow shop problem with subcontracting options and energy cost considerations is developed. To the best of our knowledge there is no comparable model in literature. To allow subcontracting in a hybrid flow shop problem, external production capacity is seen as further parallel machines in the hybrid production system. The energy costs are assumed to be time-depending. A cost-oriented objective function is implemented to minimize charges for transportation, production and energy simultaneously. CPLEX is applied to a numerical example to illustrate the presented approach.

Acknowledgement

Published Paper: S. Schulz / S. Apelmeier / U. Buscher (2017): Hybrid Flow Shop Scheduling with Subcontracting Options and Time-Depending Energy Costs. In: *Logistikmanagement - Beiträge zur LM 2017.* Ed. by R. O. Large et al. Stuttgart, pp. 163 – 170.

7.1 Introduction

Today's production industry faces a competitive environment in which flexibility and reactivity are essential factors for most companies. In times of globalization, just-in-time production and heavily fluctuating demand companies are forced to prevent bottlenecks. Firms often counteract with a growing number of temporary workers which can be very expensive. Subcontracting offers not only the advantage to continuously adjust production capacity to current demand, but also the possibility to increase capacity utilization due to reduction of idle times in different stages.[1] Another important aspect is the increased ability to meet due dates.

Although there are a few good reasons for subcontracting it is hardly considered in machine scheduling literature. This paper introduces a MIP formulation for a HFS problem with subcontracting options. Since energy costs can account for up to 60 % of production costs in energy intensive industries like chemical manufacturing,[2] the model also incorporates energy costs, allowing the energy prices to vary depending on time. If costs for external manufacturing are substantially constant, subcontracting might be especially lucrative in times of high energy prices.

Before the model formulation is given in section 7.3, the following section surveys the literature. In section 7.4 a numerical example serves to illustrate how the model operates. Finally, conclusions are given and deduced issues for further research are described.

7.2 Previous research

Subcontracting is the external allocation of those tasks exceeding companies' technical capacity. A distinction should be drawn from the term "outsourcing" which describes handing over complete tasks to a provider that can't be done in-house.[3] Some of the following articles use both terms synonymously.

Undoubtedly, various research has been made in the field of subcontracting,[4] but within the field of scheduling there are only a few articles considering external allocation. Lee / Choi (2011)[5] investigate outsourcing in a two-stage flow shop problem. Chen / Li (2008)[6] examine a parallel machine problem where jobs can be subcontracted completely,

[1] Qi (2009): *Scheduling with an option of outsourcing.*
[2] IEA (2007): *Tracking Industrial Energy Efficiency.*
[3] Chen / Li (2008): *Scheduling with subcontracting options.*
[4] See e.g. Dolgui / Kovalev / Pesch (2015): *Virtual business planning problem.*
[5] Lee / Choi (2011): *Two-stage production scheduling with an outsourcing option.*
[6] Chen / Li (2008): *Scheduling with subcontracting options.*

but the transfer of individual operations is not possible. Hong / Lee (2016)[7] depict a single machine problem with several outsourcing providers. The mentioned articles do not schedule external jobs. In our examination each operation can be swapped out individually, which requires an exact termination of subcontracted jobs. An example for such a two stage flow shop problem can be found in Li / Luan / Qiu (2016)[8]. A two stage flow shop is also analyzed in Qi (2011)[9], who examines three different outsourcing scenarios.

With due caution, we can say that subcontracting in the form considered here has not yet been examined in HFS literature. The basic idea that jobs do not have to be processed at all stages in-house is also transcribed in so-called not all machine (NAM) problems. In this context Gerstl / Mosheiov (2014)[10] describe a two-stage NAM HFS with parallel identical machines at the second stage and setup times between batches. Similarly Lei / Guo (2016)[11] investigate the HFS with NAM option.

While subcontracting options are comparatively rare in scheduling literature, EAS studies enjoy increasing popularity. A comprehensive overview about energy considerations in scheduling is given by Gahm et al. (2016)[12]. Generally speaking, there are two main strategies to take energy into account. A major part of articles tries to reduce the energy consumption directly. Hereto, different machine states (e.g. idle, standby, off) shall be optimized, production speed can be varied or heterogeneity of parallel machines with respect to energy consumption can be exploited. But costs may also be reduced while the consumption remains constant. To make this possible, volatile energy prices must be exploited, peak consumption should be reduced and special costs (e.g. load tracking errors or network charges) need to be decreased.

A MIP for EAS is formulated by Bruzzone et al. (2012)[13] to reduce peak power consumption on the basis of an APS-system. Luo et al. (2013)[14] describe an ant colony optimization to optimize makespan and energy costs in an HFS under consideration of variable energy prices and changeable machine speed. Machine state optimization like on-/off-decisions in HFS can also be found.[15]

In the next section both described ideas (subcontracting and EAS) will be combined in one MIP for the first time. The energy consumption may be reduced by making use of

[7] Hong / Lee (2016): *Outsourcing decisions in single machine scheduling.*
[8] Li / Luan / Qiu (2016): *Two-Stage Flowshop Scheduling with Outsourcing.*
[9] Qi (2011): *Outsourcing and production scheduling.*
[10] Gerstl / Mosheiov (2014): *The optimal number of used machines.*
[11] Lei / Guo (2016): *Hybrid flow shop scheduling with not-all-machines options.*
[12] Gahm et al. (2016): *Energy-efficient scheduling.*
[13] Bruzzone et al. (2012): *Energy-aware scheduling.*
[14] Luo et al. (2013): *Hybrid flow shop scheduling.*
[15] See e.g. Dai et al. (2013): *Energy-efficient scheduling*; Mashaei / Lennartson (2013): *Energy Reduction in a Pallet-Constrained Flow Shop.*

parallel machines with different consumption rates. Furthermore, time-depending energy prices make it possible to reduce energy costs while total consumption remains unchanged. These varying prices are likely to influence subcontracting decisions.

7.3 Mathematical formulation

In this section a MIP formulation for a hybrid flow shop scheduling problem with subcontracting options and energy cost considerations shall be introduced for which the following notation is adopted.

Indices (Enumerator: small letter)

E_i	Set of b_i external machines
I	Set of s stages
J	Set of n jobs
K_i	Set of o_i in-house machines
L_i	Set of all (o_i+b_i) machines
T	Set of T_{max} time periods

Parameters

a_{il}	Machine-hour rate for machine l at stage i
c_i	Transportation charge at stage i
e_t	Energy cost at time t (real time price)
p_{ijl}	Processing time of job j at stage i on machine
v_ik	Energy demand of machine k at stage i
z_i	Transportation time at stage i in case of subcontracting

Decision Variables

$C_{ij} \in \mathbb{N}$	Completion time of stage i of job j
$X_{ijlt} \in \{0,1\}$	Binary that equals 1 if job j is produced on machine l at stage i in time period t
$Y_{ijl} \in \{0,1\}$	Binary that equals 1 if job j is manufactured on machine l at stage i
$Z_{ijlt} \in \{0,1\}$	Binary that equals 1 if job j is started on machine l at stage i in time period t

7.3.1 Problem description and assumptions

The classical flow shop scheduling problem is characterized by a set of n jobs which have to be processed on at least 2 production stages, whereby all jobs have the same processing order. In a hybrid problem there is at least one stage with more than one machine.[16]

This basic problem is extended by the following assumptions:

- Parallel Machines are heterogeneous.
- Each job j has an unrelated processing time p_{ijl} at stage i on machine l.
- In-house jobs consume a fixed amount of energy v_{ik} per time period. The energy price e_t is time-depending.
- All jobs could technically be subcontracted at each stage i to at least one external subcontractor.
- In the case of subcontracting, costs and time usage are not only caused by the production itself but also by transportation between manufacturer and subcontractor. The efforts for transportation depend on the chosen external partner.
- In-house machines do not consume energy in idle times (they are turned off).
- All jobs and machines are available at time zero and each machine can process at most one job at a time. Preemption is not allowed.

Based on this verbal definition the problem will be formally described in the following.

7.3.2 Mixed integer problem formulation

Depending on the selected type of decision variables, one can distinguish between four types of MIP formulations in scheduling.[17] The proposed model is based on time indexed variables that are commonly used to model scheduling problems. The time index is favourable to model time-dependent energy costs while another formulation is more efficient in absence of this dependency.

To consider external manufacturers customarily two separated variables are used for in-house and external manufacturing.[18] We want to propose an alternative approach. Since a hybrid flow shop problem is characterized by different machines at each stage it is possible to also include external machines in this procedure. Our idea is to treat a subcontractor like a further parallel machine. Based on this idea the number of parallel machines l_i at

[16] For more information about HFS see e. g. Ruiz / Vazquez-Rodriguez (2010): *The hybrid flow shop scheduling problem.*

[17] See e.g. Keha / Khowala / Fowler (2009): *Mixed integer programming formulations.*

[18] See e.g. Chen / Li (2008): *Scheduling with subcontracting options.*

each stage can be subdivided into internal machines o_i and external machines b_i. Following this logic, we can formulate the constraints below.

$$\sum_{t=1}^{T_{max}} \sum_{k=1}^{o_i} \frac{X_{ijkt}}{p_{ijk}} + \sum_{e=o_i+1}^{o_i+b_i} Y_{ije} = 1 \quad \forall i \in I, j \in J \tag{7.1}$$

$$\sum_{t=1}^{T_{max}} \frac{X_{ijkt}}{p_{ijk}} = Y_{ijk} \quad \forall i \in I, j \in J, k \in K_i \tag{7.2}$$

$$\sum_{t=1}^{T_{max}} \frac{X_{ijet}}{p_{ije} + z_i} = Y_{ije} \quad \forall i \in I, j \in J, e \in E_i \tag{7.3}$$

$$\sum_{j=1}^{n} X_{ijlt} \leq 1 \quad \forall i \in I, l \in L_i, t \in T \tag{7.4}$$

$$C_{ij} \leq (1 + X_{ijlt+1} - X_{ijlt}) \cdot T_{max} + t \quad \forall i \in I, j \in J, l \in L_i, t \in T | t < T_{max} \tag{7.5}$$

$$C_{ij} \geq X_{ijlt} \cdot t \quad \forall i \in I, j \in J, l \in L_i, t \in T \tag{7.6}$$

$$C_{ij} \geq C_{i-1,j} + \sum_{t=1}^{T_{max}} \sum_{k=1}^{o_i} X_{ijkt} + \sum_{e=o_i+1}^{o_i+b_i} Y_{ije} \left(p_{ije} + z_i\right) \quad \forall i \in I | i > 1, j \in J \tag{7.7}$$

$$C_{1j} \geq C_{1,j} + \sum_{t=1}^{T_{max}} \sum_{k=1}^{o_1} X_{1jkt} + \sum_{e=o_1+1}^{o_1+b_1} Y_{1je} \left(p_{1je} + z_1\right) \quad \forall j \in J \tag{7.8}$$

$$\sum_{t=1}^{T_{max}} \sum_{l=1}^{o_i+b_i} Z_{ijlt} \cdot t = C_{ij} - \sum_{t=1}^{T_{max}} \sum_{k=1}^{o_i} X_{ijkt} - \sum_{e=o_i+1}^{o_i+b_i} Y_{ije} \left(p_{ije} + z_i\right) + 1 \quad \forall i \in I, j \in J \tag{7.9}$$

$$\sum_{t=1}^{T_{max}} Z_{ijlt} = Y_{ijl} \quad \forall i \in I, j \in J, l \in L_i \tag{7.10}$$

$$Z_{ijlt} \geq X_{ijlt} - X_{ijlt-1} \quad \forall i \in I, j \in J, l \in L_i, t \in T | t > 1 \tag{7.11}$$

$$Z_{ijl1} \geq X_{ijlt} \quad \forall i \in I, j \in J, l \in L_i \tag{7.12}$$

Constraint (7.1) defines that each task of each job is completed either in-house or extern. We introduce (7.2) and (7.3) to accurately reflect the processing time needed as well as transportation time if necessary. The combination of equations (7.1), (7.2) and (7.3) ensures that each job is assigned to exactly one machine at each stage. Consequently, it also implements the impossibility for a job to skip a stage. At each time interval t each job can be allocated at most once, which is specified by (7.4). The completion time C_{ij} is calculated in constraints (7.5) and (7.6). On this basis, equation (7.7) guarantees that a job cannot be processed on a stage before the previous stage is finished respectively a subcontracted job is returned. Condition (7.8) is not necessary to define the problem clearly, but reduces the solution space and consequently the optimization time. We assume non-preemption (tasks cannot be interrupted) and hence we calculate in equation (7.9) a starting time period Z_{ijlt}. With (7.10) it is ensured that every job starts only once which logically eliminates possible interruptions. Finally, constraints (7.11) and (7.12) connect the variables Z_{ijlt} and X_{ijlt}. To complete the model formulation the objective function is introduced in the next section.

7.3.3 Objective function

Finding a suitable objective function that appropriately represents the real target can be difficult, especially if different dimensions have to be covered. To consider production time, energy consumption and transportation simultaneously, a cost oriented approach is recommended. With the aid of the subcontracting decision, the objective function can be subdivided into two parts.

The internal manufacturing costs include energy expenses and the machine caused shares. Thereby, energy costs derive from real-time prices at the stock exchange (e_t) multiplied by the actual machine run time and the corresponding energy consumption rate (v_{ik}). To calculate the production costs, a machine-hour rate (a_{ik}) is used. This factor represents manufacturing costs like depreciation and personnel expenses but excludes energy costs.

The external costs accordingly contain the processing costs (a_{ie}) multiplied by the processing time (p_{ije}). Additionally, they include the product of transportation cost rate (ci) and transportation time (zi) which both depend on the respective subcontractor. Combining internal and external costs leads to the following objective function:

$$\sum_{j=1}^{n}\sum_{i=1}^{s}\left(\sum_{t=1}^{T_{max}}\sum_{k=1}^{o_i} X_{ijkt}\cdot(e_t\cdot v_{ik}+a_{ik})+\sum_{e=o_i+1}^{o_i+b_i} Y_{ije}\cdot(p_{ije}\cdot a_{ie}+z_i\cdot c_i)\right)\Rightarrow Min \quad (7.13)$$

7.4 Numeric example

A numerical example is introduced in this section to provide a transparent presentation of how the model operates. We consider a two-stage hybrid flow shop with two parallel machines as well as one external subcontractor on each stage. Over a period of 24 hours 10 jobs shall be processed. To allow the energy costs to be as realistic as possible, Phelix spot market prices are used from 10th January 2017 (see figure 7.1). All other parameters are generated randomly by using the following discrete uniform distributions.

- Processing time p_{ijl} [h]: $unif\{2;6\}$
- Energy demand v_{ik} [kWh]: $unif\{100;500\}$
- Transportation time z_i [h]: $unif\{1;3\}$
- Transportation charge c_i [€/h]: $unif\{30;70\}$
- Internal machine cost $a_{ik} \forall k \in K_i$ [€/h]: $unif 100;200$
- External machine cost rate $a_{ik} \forall k \in E_i$ [€/h]: $\lfloor max_{k \in K_i}(a_{ik} \cdot 1.2) \rfloor$

To show that low energy cost proportions may also influence the schedule, mean energy costs are assumed to account for approximately 10 % of in-house manufacturing costs. External manufacturing costs are 20 % above the most expensive in-house machine costs. Following both assumptions, subcontracting expenses are, on average, 10 % higher than in-house efforts. Transportation costs enlarge this gap.

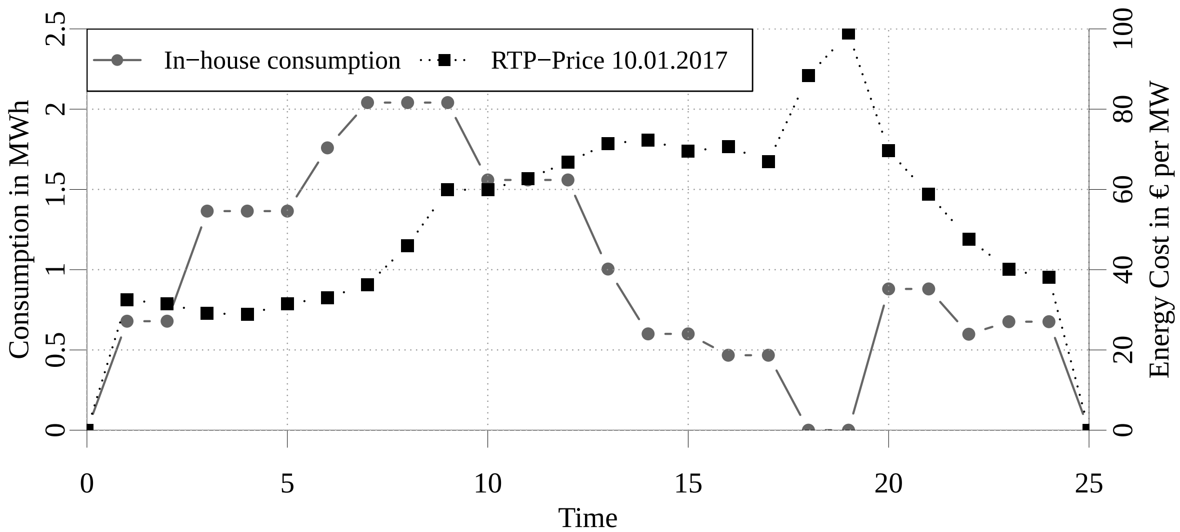


Figure 7.1: Considered energy prices and load curve of the optimal solution

The described problem is solved by using CPLEX 12.6 via OPL on an Intel Xeon, 3.46 GHz computer. We have limited the number of threads to 8 and with this, the solver needed 34.84 s on average to calculate the optimal solution of 15261.68 €. These costs can be subdivided into 11357 € internal machine costs, 1224.68 € electricity costs and 2680 € external costs including 228 € for transportation. The optimal schedule is visualized in figure 7.2.

It can be seen that, overall, four jobs are subcontracted. External manufacturing is especially used in the last stage, even though the external machine in stage 3 is the most expensive one at 240 €/h. In the first two stages only one job is subcontracted. It could be supposed that low transportation costs are the reason for outsourcing at stage 3. However,

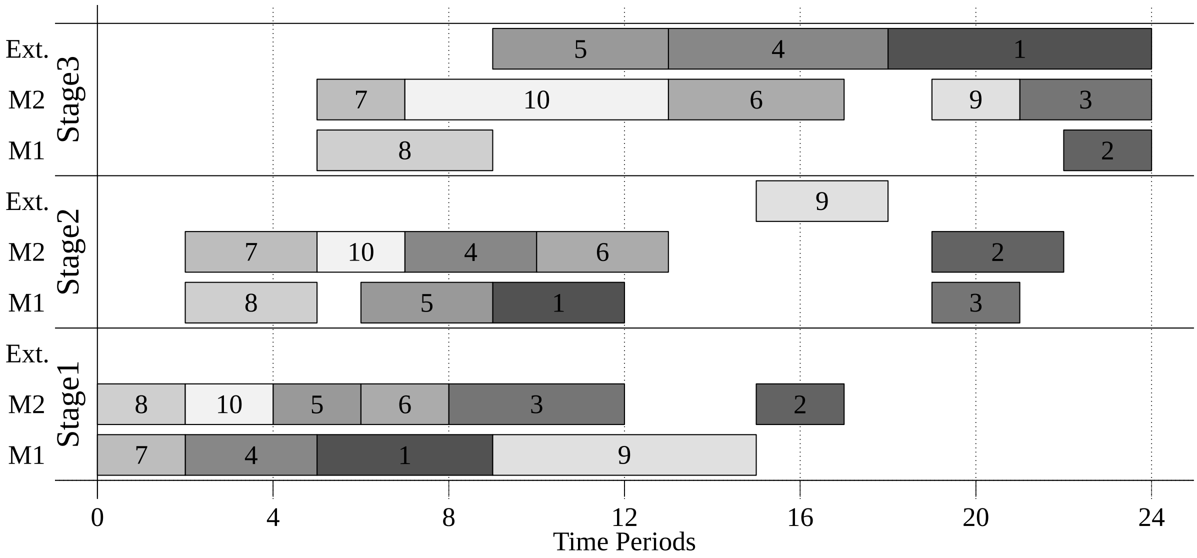


Figure 7.2: Optimal schedule for the numerical example

interestingly the transportation in stage 3 with 60 € is more expensive than in stage 1 and 2 with 36 € respectively 48 €. Thus, the cause for the relocations in stage 3 could be seen in the high energy costs during the evening. In this context, in-house production is completely avoided between 5 and 7 pm.

By neglecting subcontractors, total costs would increase by more than 7 % to 16362.13 €. This indicates that subcontracting may be a profitable supplement in scheduling. In spite of the low proportion of energy costs a certain influence can be identified. The conscious reduction of energy consumption in peak hours is visualized in figure 7.1. Energy demand and real-time prices run in opposite directions. Such preventions of peak hour consumptions not only profit the reduction of costs but also the energy network stability.

The numerical example above shows that subcontracting can help to reduce manufacturing costs. Nevertheless, the test run may indicate only main properties and is thought to hint at the influence of subcontracting and energy costs. To derive reliable statements a more detailed computational analysis is required. In this respect, the following questions could also be analyzed:

- Influence of quantity and type of subcontractors to machine scheduling
- Relationship between transport distance, costs and frequency of subcontracting
- Expensive new machines vs. inefficient old machines – impact of energy costs

Further outstanding issues shall be described in the last section.

7.5 Conclusion and outlook

This paper introduces a MIP formulation for a hybrid flow shop scheduling problem with subcontracting options and time depending energy prices. Subcontracting still draws little consideration in scheduling literature. To the best of the authors' knowledge, there is no article analyzing subcontracting in hybrid flow shop scheduling combined with energy cost considerations. It has been shown that subcontracting provides a new form of flexibility which helps to overcome increasing fluctuations of demand and growing deadline pressure. Particularly in times of high variable costs (e.g. energy expenses), subcontracting can also be useful to reduce manufacturing costs.

It goes without saying that the proposed model can be solved to optimality only for small instances which requires heuristic approaches to resolve industrial sized problems. On this account, a better suited genetic algorithm will be developed in a future work. Disregarding the capacity utilization within the objective function poses another minor concern. In subsequent studies a multi-objective approach shall be derived by introducing a second objective like makespan or total completion time.

8 A genetic algorithm to solve the hybrid flow shop scheduling problem with subcontracting options and energy cost consideration

Abstract

This paper analyses the hybrid flow shop scheduling problem (HFSSP) with subcontracting options and time depending energy costs. While the consideration of energy costs in scheduling has increased considerably in recent years, subcontracting is rarely analysed in scheduling literature. A mathematical MILP formulation is given to define the exact problem and to calculate optimal solutions for small instances. The objective is to minimise the total production costs for internal and external manufacturing including transportation and energy costs. Since, already the general HFSSP is NP-hard the considered problem is difficult to solve to optimality. Therefore, a genetic algorithm (GA) based on a detailed matrix encoding procedure is proposed. To the best of my knowledge this is the first time that a heuristic approach is presented for the considered problem. An algorithm for intelligent swaps to make use of waiting time and a right-shifting procedure to take advantage of time depending energy costs prove to be suitable to improve the performance of the GA significantly. It can be shown that the GA finds nearly optimal solutions in a very short time.

Acknowledgement

Published Paper: S. Schulz (2019): A genetic algorithm to solve the hybrid flow shop scheduling problem with subcontracting options and energy cost consideration. In: *Advances in Intelligent Systems and Computing*, vol. 854, pp. 263–273.

8.1 Introduction

Since companies have to be become increasingly flexible and networked, temporary purchase of production capacities can be an important competitive advantage. Subcontracting is the possibility to allocate a single processing step of a job to an external manufacturer called subcontractor. Different forms of outsourcing are normally analysed on a strategic management basis. For example Hahn et al. (2016)[1] examine robust outsourcing decision making. However, short-term uncertainties such as machine failures or new orders may have a decisive influence on the subcontracting decisions. Nevertheless, at the operational level and in particular in production planning and scheduling subcontracting is rather less taken into account.

Choi / Chung (2016)[2] examine outsourcing in a single machine problem with processing time uncertainty. Lee / Sung (2008)[3] consider subcontracting in a single machine layout as well and weigh between delay and outsourcing costs. Parallel machine problems combined with strategic outsourcing decisions are analysed by Chen / Li (2008)[4] as well as Mokhtari / Abadi (2013)[5]. Qi (2011)[6] presents different models for subcontracting in a two stage flow shop problem. Li / Luan / Qiu (2016)[7] describe a bi-objective flow shop problem with outsourcing possibilities, whereby different jobs are discounted at external machines while in-house machines have to be maintained. In HFSSP subcontracting is rarely considered. A mathematical formulation is introduced by Schulz / Apelmeier / Buscher (2017)[8].

Besides subcontracting this paper also considers time-depending energy costs. In the context of growing concern about environmental pollution and increasing energy costs as well as demand energy aware scheduling has received a lot of attention in recent years. Altogether more than 100 articles were published since 2010.[9] In addition to the possibility of reducing costs through intelligent planning by taking advantage of time-dependent price models or reducing peak power, scheduling can also be used to reduce the energy consumption. In principle, three approaches are conceivable. Firstly, the processing speed can be adapted to save energy in cost of longer processing times. Secondly, different machine states can by considered to reduce standby times and turn machines completely

[1] Hahn et al. (2016): *A multi-criteria approach to robust outsourcing decision-making.*
[2] Choi / Chung (2016): *Min−max regret version of a scheduling problem.*
[3] Lee / Sung (2008): *Single machine scheduling with outsourcing allowed.*
[4] Chen / Li (2008): *Scheduling with subcontracting options.*
[5] Mokhtari / Abadi (2013): *Scheduling with an outsourcing option.*
[6] Qi (2011): *Outsourcing and production scheduling.*
[7] Li / Luan / Qiu (2016): *Two-Stage Flowshop Scheduling with Outsourcing.*
[8] Schulz / Apelmeier / Buscher (2017): *Hybrid Flow Shop Scheduling with Subcontracting Options.*
[9] Cf. Schulz (2018): *A Multi-criteria MILP Formulation.*

off. Thirdly, in the case of heterogeneous parallel machines, priority can be given to more efficient machines. An overview about different existing approaches is given by Biel / Glock (2016)[10] or Gahm et al. (2016)[11].

In section 8.2 a MILP formulation is introduced to define the problem exactly. Afterwards in section 8.3 a modified GA is proposed. In section 8.4 follows a computational study analysing the performance of the heuristic compared to the optimal solution. Finally a summary as well as a short outlook are given.

8.2 Mathematical model formulation

Indices		**Parameter**	
$e \in E_i$	Set of b_i external machines	a_{il}	Machine-hour rate
$i \in I$	Set of m stages	c_i	Transportation charge
$j \in J$	Set of n jobs	e_t	Real time energy price (RTP)
$k \in K_i$	Set of o_i in-house machines	p_{ijl}	Processing time
$l \in L_i$	Set of all $(o_i + b_i)$ machines	v_{ik}	Energy demand
$t \in T$	Set of τ time intervals	z_{ie}	Transportation time

The considered problem can be formulated as a MILP. A similar model can be found in Schulz / Apelmeier / Buscher (2017)[12]. However, some adjustments have been made to make the model more compact and to speed up the calculation of the optimal solution.

Altogether n jobs must be processed at m production stages. At each stage a number of o_i in-house machines as well as b_i subcontractors are available. The different machines are heterogeneous, which means that processing time and energy demand can vary for the same task. In the model, subcontractors are initially considered as further parallel machines. The production period under consideration is divided into τ equal time intervals where processing begins in time interval 1. The notation given above is used for the model formulation. Three different **decision variables** are used. The **binary** X_{ijtk} is equal to 1, if a job j is processed at stage i on machine l in time interval t. To assign job j to machine l at stage i the **binary** Y_{ijk} becomes 1. The value of **integer** C_{ij} corresponds to the completion time of job j at stage i.

The objective function (1) minimizes the total costs for a given production period. The first part displays the in-house costs, consisting of processing and separate energy cost, where the costs are added up over the individual time periods. The second part includes

[10] Biel / Glock (2016): *Energy-efficient production planning.*
[11] Gahm et al. (2016): *Energy-efficient scheduling.*
[12] Schulz / Apelmeier / Buscher (2017): *Hybrid Flow Shop Scheduling with Subcontracting Options.*

fixed external production costs independent of time as well as transport costs.
Minimize:

$$\sum_{j\in J}\sum_{i\in I}\left[\sum_{t\in T}\sum_{k\in K_i} X_{ijkt}\left(e_t\cdot v_{ik}+a_{ik}\right)+\sum_{e\in E_i} Y_{ije}\left(p_{ije}\cdot a_{ie}+z_{ie}\cdot c_i\right)\right] \tag{8.1}$$

Subject to:

$$\sum_{t\in T}\sum_{k\in K_i}\frac{X_{ijkt}}{p_{ijk}}+\sum_{e\in E_i} Y_{ije}=1 \quad \forall i\in I, j\in J \tag{8.2}$$

$$\sum_{t\in T} X_{ijkt}=Y_{ijk}\cdot p_{ijk} \quad \forall i\in I, j\in J, k\in K_i \tag{8.3}$$

$$\sum_{t\in T} X_{ijet}=Y_{ije}\left(p_{ije}+z_{ie}\right) \quad \forall i\in I, j\in J, e\in E_i \tag{8.4}$$

$$\sum_{j\in J} X_{ijlt}\leq 1 \quad \forall i\in I, l\in L_i, t\in T \tag{8.5}$$

$$\sum_{l\in L_i}\sum_{t\in T|t>1}\left|X_{ijlt}-X_{ijlt-1}\right|+X_{ijl1}+X_{ijl\tau}=2 \quad \forall i\in I, j\in J \tag{8.6}$$

$$C_{ij}\leq X_{ijlt}\cdot t \quad \forall i\in I, j\in J, l\in L_i, t\in T \tag{8.7}$$

$$C_{ij}\leq (1+X_{ijlt+1}-X_{ijlt})\tau+t \quad \forall i\in I, j\in J, l\in L_i, t\in T|t<\tau \tag{8.8}$$

$$C_{ij}\geq C_{i-1,j}+\sum_{t\in T}\sum_{k\in K_i} X_{ijkt}+\sum_{e\in E_i} Y_{ije}\left(p_{ije}+z_{ie}\right) \quad \forall i\in I|i>1, j\in J \tag{8.9}$$

$$C_{ij}\geq \sum_{i^*=1}^{i}\sum_{k\in K_{i^*}} X_{i^*jkt}\cdot t+\sum_{e\in E_{i^*}} Y_{i^*je}\left(p_{i^*je}+z_{i^*e}\right) \quad \forall i\in I, j\in J, t\in T \tag{8.10}$$

$$C_{mj}\leq \tau \quad \forall j\in J \tag{8.11}$$

With constraint (8.2) each job is assigned to an in-house or external machine and it is ensured that a job does not skip a stage. Equation (8.3) and (8.4) assign a job to a machine for the entire processing and transport time if necessary. Condition (8.5) specifies that a machine can process a maximum of one job at a time. Since non-preemption is assumed, equation (8.6) is introduced to avoid interruptions. In (8.7) and (8.8) the completion time C_{ij} is defined depending on X_{ijlt}. Based on that value constraint (8.9) ensures that a job cannot be started until it has been completed at the previous stage of production. The last two inequalities (8.10) and (8.11) are not necessary to define the problem, but reduce the solution space and thus accelerate the solution finding of the solver. The basic idea is to limit the range of C_{ij} which saves up to 85% solution time for some of the test instances. To solve the model IBM ILOG CPLEX 12.6 is used.

8.3 Genetic algorithm

Since the problem is only solvable for small instances to optimality a heuristic approach is necessary for industrial size problems. For that reason a Genetic algorithm shall be suggested here. The basic procedure can be seen in figure 8.1. A detailed description is given below.

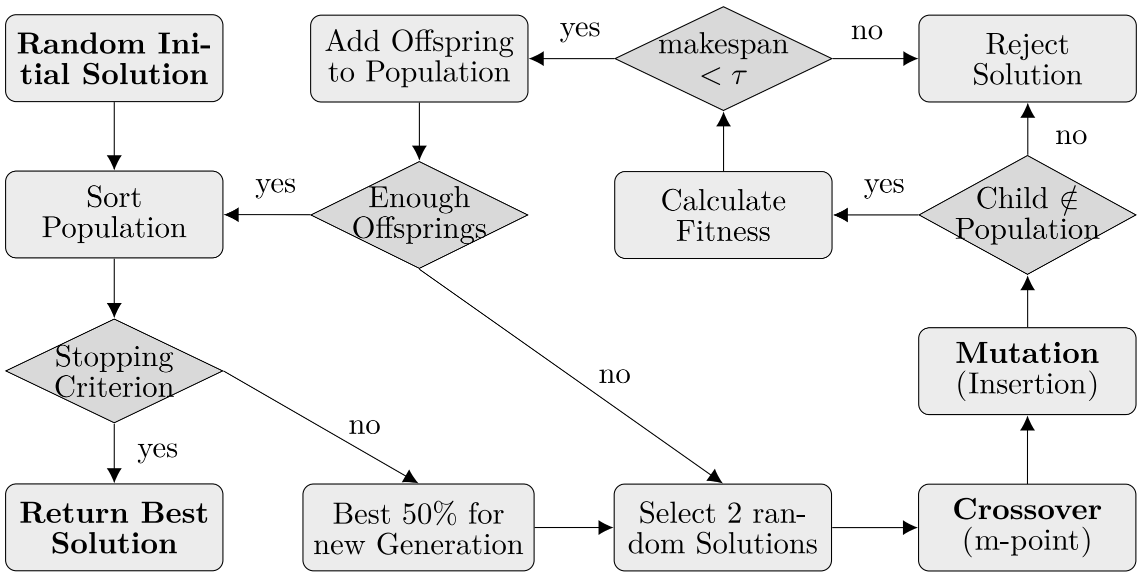


Figure 8.1: General procedure of the proposed genetic algorithm

8.3.1 General procedure

Decoding & encoding:

In contrast to the classic flow shop problems, in HFSSP not only the sequence has to be encoded, but also the machine assignment influences the quality of a solution and must therefore be encrypted. It is common that only the sequence at first production stage is used to represent the solution. Afterwards the schedule is generated by list scheduling algorithms.[13] This procedure often proves to be advantageous, especially in the case of larger problem instances. However, many solutions are excluded with this approach. For that reason, a matrix coding procedure shall be used at this point, which contains both sequence and machine assignment in all production levels. This procedure seems to be more suitable for problems with a small number of production stages. A similar approach is used by Dai et al. (2013)[14].

[13] See e.g. Ruiz / Maroto (2006): *A genetic algorithm for hybrid flowshops.*
[14] Dai et al. (2013): *Energy-efficient scheduling.*

An example for 4 jobs and 2 production stages can be seen in (8.12). A decimal number is assigned to each job at each stage. The integer part assigns a machine to the Job. For example, in the first stage Job 1, 3 and 4 are processed on machine 1. The decimal places determine the sequence. The smaller this value, the earlier the job is processed. This means that in the given example the order on machine 1 at stage 1 is 3-1-4. If two jobs have the same priority the job number is used for ordering. For larger instances three decimal places are considered.

$$\begin{array}{c c} & \begin{array}{cccc} Job\ 1 & Job\ 2 & Job\ 3 & Job\ 4 \end{array} \\ \begin{array}{c} Stage\ 1 \\ Stage\ 2 \end{array} & \begin{pmatrix} 1.56 & 2.42 & 1.33 & 1.67 \\ 2.12 & 1.89 & 2.89 & 1.45 \end{pmatrix} \end{array} \tag{8.12}$$

Initial solution:

To start the GA firstly a start population is needed. Since the calculation of a constructive solution takes some time and the influence within a larger population is estimated relatively low, we generate random solutions. Makespan is not an objective for the considered problem but nonetheless there is a very limited period of production (τ). The used coding procedure considers all possible solution. But with this also a lot of invalid solutions are possible. To avoid considering to many of these solutions and to accelerate the solution finding the priority values of the first stage are used for all following stages. Thus, there are $n!$ possibilities. However, the assignment to machines is completely random, which leads to $n! \cdot \prod_{i \in I}(L_i)^n$ possible initial solutions, what is the upper bound for the population size.

Crossover and mutation:

After the start population is sorted the best 50% are selected for a new population which can be seen in figure 8.1. Within the new generation, two parents are selected randomly. They produce two offspring by recombination. The recombination consists of crossover and mutation. The individuals are crossed at a different point in each production stage. These m points $\alpha_S \in \{0, .., n\}$ are generated randomly. An example can be seen in figure 8.2. The first selected parent passes the information for the first α_s jobs at stage s to child 1 and the rest to Child 2. The missing information of the offspring comes from parent 2.

Afterwards, each generated offspring is modified with a mutation probability η_m at one point. This means that a randomly selected job is either assigned to another machine or receives a different priority at one production stage.

Parent 1

1.56	2.42	1.33	1.67
2.12	3.89	2.89	1.45

+

Parent 2

1.02	1.27	2.45	2.89
3.99	2.25	2.65	3.13

=

Child 1

1.56	2.42	1.33	2.89
2.12	2.25	2.65	3.13

Child 2

1.02	1.27	2.45	1.67
3.99	3.89	2.89	1.45

Figure 8.2: m-point crossover procedure for $\alpha_s = \{3; 1\}$

Evaluation and stopping criterion:

After a offspring is created, firstly it is checked if the same individual is not already part of the population. Identical solutions are rejected to maintain diversification. As can be seen in figure 8.1, the costs for new solutions, which are also used as fitness, are then calculated. Schedules that exceed the given maximum processing time τ are rejected at this point. All other solutions are included in the new population. When the new generation contains enough individuals the solutions are sorted and the procedure restarts. The algorithm is terminated if the best solution has not changed within γ generations, where γ depends on the size of the problem instance. After termination the best result is returned.

8.3.2 Adjustments for improvement

Two major properties of the described problem can be identified in initial tests. Based on these findings the following two algorithms are implemented to improve the solution quality and accelerate the process.

I. Intelligent swaps:

The coding procedure enables jobs that are processed last in a stage to be scheduled very early in a subsequent stage and vice versa. This may result in long waiting and idle times. In turn it may lead to high makespan and thus to invalid solutions. We integrate intelligent swaps in the algorithm to process waiting jobs in idle times.

Simply said it is tested if a job can be scheduled before the previous job on a machine. For example in figure 8.3 it is tested if the idle time between job 3 and job 4 at stage 2 is enough to process job 2 and whether job 2 is completed at stage 1 before it should begin on stage 2. Since both requirements are met, job 2 and 4 are swapped and makespan can be reduced from 17 to 13. To swap two jobs their priority values are exchanged. Theoretically, it could also be beneficial if a job overtakes two or more jobs simultaneously, but here swaps are just tested for the direct predecessor to limit the computational effort. However, it is possible that several jobs overtake the same predecessor.

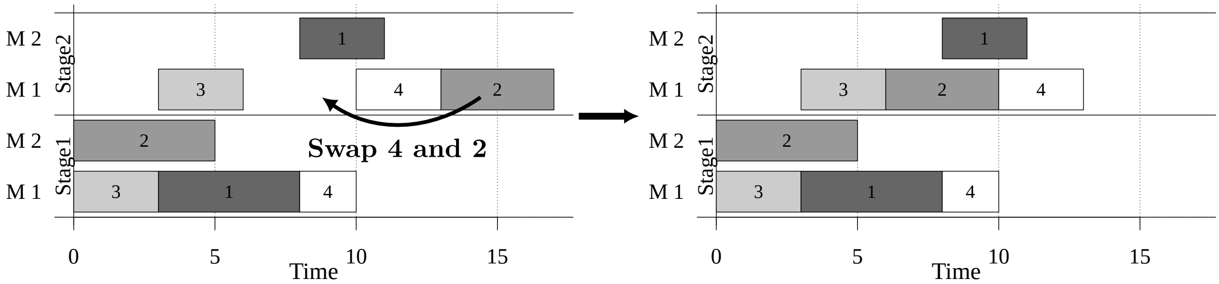


Figure 8.3: Visualization of intelligent swaps

II. Right shifting procedure:

Energy costs are not initially taken into account in the planning. Jobs are always scheduled as early as possible. With a right shifting algorithm it is tested whether the energy costs can be reduced by later processing. Therefore, beginning with the last stage the last job on a machine is stepwise shifted right until maximum completion time is reached. For each postponement the reduced costs are calculated and the best position is fixed. Subsequently the previous jobs are shifted stepwise until it finishes immediately before the next job starts. For earlier stages it must also be considered that the job must be done before processing starts at the next stage. Of course the right shifting can only be carried out for in-house production.

8.4 Computational study

To analyse the performance of the introduced GA computational tests shall be examined. The results are compared with the optimal solution, which is calculated using the model in section 8.2. For all tests an Intel Xeon 3.3 GHz CPU with 768 GB memory is used.

8.4.1 Test instances

Since the problem under consideration has not yet been addressed in literature, there are no existing test instances. Thus, new examples are generated at this point. Altogether we consider 84 different instances. The various combinations of number of jobs, production stages as well as in-house and external machines are listed in table 8.1. Problems with 50 and 100 jobs can be solved in around 10 respectively 30 seconds by the GA. Since

CPLEX cannot find solutions for most of these instances, the quality of the results cannot be evaluated and therefore they are not given in detail here.

The data are randomly generated in the areas shown in table 8.2. In order to be able to solve even larger instances to optimality relatively short processing times are taken into account. It is assumed that average energy costs account for approximately 10% of internal costs. In energy intensive industries like chemical industries the share can be up to 80% of production costs.[15] For energy prices we use Phelix spot market prices from 16th of March, 2017. External production costs are 20% higher than those of the most expensive internal machine. Added to this are the transport costs and times, which initially make external production appear significantly more expensive.

Set	**Size**
n (Jobs)	6,8,10,12,15,20,30
m (Stages)	2,3,4
o_i (In-house)	2,3
b_i (External)	1,2

Table 8.1: Problem sizes

Parameter	**Range**
Processing time $p_{ijl}[h]$	$unif\{1, 10\}$
Energy demand $v_{ik}[kWh]$	$unif\{100, 1000\}$
Transportation time $z_{ie}[h]$	$unif\{1, 3\}$
Transportation charge $c_i[€/h]$	$unif\{30, 70\}$
Internal machine cost $a_{ik}[€/h]$	$unif\{100, 200\}$
External costs $a_{ie}[€/h]$	$\lfloor \max_{k \in K_i}(a_{ik}) \cdot 1.2 \rfloor$

Table 8.2: Overview of the test data

Furthermore, the maximum completion time largely influences the scheduling procedure. It should be possible that all jobs can be produced in-house and simultaneously the time horizon must not be too high to produce only in times of low electricity prices. The term 8.13 is used to calculate the maximum completion time for each instance. Logically, the production can be completed earlier.

$$\tau = \left\lfloor \frac{\sum_{i \in I} \sum_{j \in J} \max_{k \in K_i}(p_{ijk})}{\sum_{i \in I} o_i} \right\rfloor + (m-1) \cdot \max_{i \in I, j \in J, k \in K_i}(p_{ijk}) \tag{8.13}$$

8.4.2 Numerical results

While CPLEX optimizes each of the 84 problems exactly once, we run the GA for each instance 30 times to analyse the scatter in terms of the solution quality and computing time. To run the GA firstly the **parameters** have to be set. Different computational tests were carried out for this purpose. A key finding is that because of the same sequence at all stages in the initial solutions, a high mutation rate proves to be advantageous. Furthermore a problem-dependent stopping criteria seems to have a valuable influence on

[15] See e.g. IEA (2007): *Tracking Industrial Energy Efficiency.*

the performance. Due to the limited scope of this article, we waive detailed explanations at this point. Overall the following settings are made:

- Population size: 100
- Mutation probability (η_m): 0.9
- Generations with unchanged best solution (γ): $10 + 5 \cdot n$

The results of the numerical study are shown in table 8.3. The 84 test instances are grouped once according to the amount of jobs, once according to the number of stages and once according to type of machines in order to identify effects.

	Optimal solution			**Genetic Algortihm**				
	Costs[€]	Gap[%]	CPU[s]	Costs[€]	SD[%]	CPU[s]	SD[%]	DTO[%]
Results depending on number of jobs								
n=6	8820.56	-	4.86	8853.61	0.02	0.08	17.47	0.31
n=8	11806.59	-	11.52	11856.82	0.03	0.15	18.29	0.31
n=10	13858.40	-	70.96	13937.65	0.07	0.27	22.94	0.44
n=12	17630.41	-	108.56	17714.72	0.05	0.40	22.39	0.39
n=15	19876.05	-	389.95	19984.00	0.08	0.74	25.74	0.37
n=20	28787.24	0.03	1146.40	28943.58	0.08	1.53	26.25	0.34
n=30	42647.98	0.11	2324.68	42905.19	0.12	4.52	30.70	0.38
Results depending on number of production stages								
m=2	13513.93	-	53.75	13555.16	0.03	0.56	21.86	0.21
m=3	19913.97	0.01	506.95	20033.12	0.07	1.09	22.83	0.43
m=4	28040.91	0.05	1177.99	28209.82	0.09	1.64	25.49	0.45
Results depending on number of in-house machines and subcontractors								
o_i=2|b_i=1	24105.41	0.02	626.79	24239.78	0.08	1.03	27.02	0.37
o_i=2|b_i=2	20418.56	0.01	462.20	20491.25	0.04	1.10	21.85	0.21
o_i=3|b_i=1	18953.95	0.03	737.03	19076.72	0.08	1.08	22.06	0.41
o_i=3|b_i=2	18480.49	0.01	492.23	18589.72	0.06	1.18	22.66	0.46
Total	**20489.61**	**0.02**	**579.56**	**20599.37**	**0.06**	**1.10**	**23.40**	**0.36**

Notation: CPU - Computation time; SD - Standard Deviation; DTO - Distance to Optimum

Table 8.3: Numerical results

For the **optimal solution**, the minimum costs and the average calculation times are shown. The calculation time of the solver is limited to 1 h, which means that some instances with 20 or 30 jobs are not optimally solved. The average distance to the lower bound is shown in the Gap column. Logically, the problem is harder to solve with an increasing number of stages which can also be seen in CPU time. In contrast to the classic HFSSP, the computing time does not decrease with an increase in parallel internal machines. However, more external machines seem to make the problem easier to solve.

With regard to costs, there is logically an increase with more jobs or stages. Additional machines reduce costs, with another in-house machine having a greater impact, as internal production is on average cheaper.

The average minimum costs found and the calculation time (CPU) are also given for the **GA**. Furthermore, the relative standard deviation (SD) is given for both values. Since the GA is executed 30 times for each instance and the algorithm is partly influenced by chance the results may scatter. With regard to the best objective found, the algorithm always finds similarly good solutions and there seem to be hardly any outliers. Overall, the results scatter by only 0.06%. The calculation time is quite different. Here there is an average of 23.4% deviation. The reason for these large differences in the calculation time for the same problem lies mainly in the stopping criterion. The algorithm ends if the best solution has not changed for γ generations. If a good solution is found in the beginning and can not be improved for γ generations the algorithm can be stopped very early. Vice versa it can take time if many smaller improvements are achieved.

When the results are compared, it can first be determined that the GA is significantly faster. The heuristic approach needs 1.1 s while the solver takes almost 10 minutes in average. Nevertheless, the results of the GA deviate only slightly from the optimal solution. The last column DTO shows the relative deviation of the best solution found for an instance from the optimum. This means that the implemented approach deviates from the optimum by only 0.36%. Especially for small problem sizes, the approach often even finds the optimal solution. These values also show that the difference remains at a similar level with an increase in the number of jobs as the termination criterion is job-dependent. With more production stages the performance of the GA slightly declines. Possibly the number of stages should also be taken into account when selecting parameters.

8.5 Summary and outlook

This paper analyses the influence of subcontracting and time depending energy prices on HFSSP. It can be shown that the temporary purchase of production capacities can be beneficial and reduce the total production costs. Especially in time of high capacity utilizations as well as high energy costs the production schedule can be improved by intelligent subcontracting.

To analyse and solve the problem a MILP formulation is given and a GA is proposed. The heuristic approach is based on a matrix encoding procedure which considers all possible schedules. To improve the performance besides basic adoptions in crossover and mutation two improvement algorithms are included. Firstly, intelligent swaps are examined to make use of waiting times. Secondly, a right shifting procedure makes use of the fluctuating energy prices. With these adoption the GA seems to be highly suitable.

In the proposed model the production time horizon is assumed to be a given value. In a future work, the makespan shall be taken into account in a multi-objective approach. Thus, depending on the situation, a production period with associated minimum costs can be selected using the Pareto front. Furthermore, subcontracted jobs could be exactly scheduled to increase the possibilities of subcontracting.

9 Conclusion and outlook

9.1 Summary and discussion of the research questions

Companies worldwide are confronted with rising energy costs. German manufacturing industries spent 32.2 billion Euro on energy in production in 2017. Fifteen years ago in 2002, energy costs were at 20.5 billion Euro.[1] Consequently, companies are trying to reduce energy consumption and costs. This can not only improve economic efficiency, but resulting reductions of environmental pollution can also improve public awareness and thus act as a marketing instrument. To reduce energy demand, companies invest in more efficient technologies and processes. Such projects, however, require high investments.

This thesis has shown that targeted operational planning can actively influence energy consumption and costs. The main interest was to investigate the impact that electricity cost consideration has on scheduling decisions. Thereby, this work focuses on three specific hybrid flow shop problems which show practical relevance and have not been discussed in literature yet All three problems were first clearly described by means of MIP formulations. Due to their complexity, larger and thus practice-relevant problems could only be solved heuristically. Therefore, a core element of this work was the development of efficient solution algorithms. HPSO, ILS and GA have proven to be suitable for the respective problems. Furthermore, existing approaches from the literature like the NSGA-II and different PSO methods were used and implemented for the evaluation.

In the field of EAS, several approaches are proposed to reduce energy consumption and costs. Research question **Q1**[2] asks for an overview of existing ideas. Basically, a distinction can be made between methods that reduce energy consumption directly and strategies that reduce costs at constant consumption. Energy demand can be reduced with scheduling by:

- Speed reductions at the expense of longer processing times (Chapter 3 & 4),
- greater utilization of more efficient parallel machines (Chapter 5, 6, 7 & 8),
- or considering machine states like idle, on, off, standby (not examined in detail).

[1]Cf. BMWI (2019): *Gesamtausgabe der Energiedaten*, p. 27.
[2]Q1: What are the main approaches in energy aware scheduling and what is the current state of research?

Switching off larger production machines only makes sense during longer idle periods due to the increased consumption when switching off and on. Generating such long idle times during scheduling contradicts the basic idea of high machine utilization for efficient production. Therefore, this approach was not pursued further in this thesis. The other two strategies have been examined intensively. Likewise, three possibilities can be identified to reduce electricity costs while maintaining a constant level of consumption:

- Exploiting off-peak price windows in time-of-use contracts (Chapter 3 & 4),
- utilisation of fluctuating real time electricity prices (Chapter 5, 6, 7 & 8),
- or reducing demand charge costs through peak power levelling (Chapter 5 & 6).

Overall, an analysis of the literature has shown that most EAS approaches concentrate on multi-criteria problems. Thereby, a large proportion takes makespan into account. Primarily heuristic and especially metaheuristic approaches are proposed, which may also be due to the multi-criteria optimization. Time-oriented objectives such as total tardiness or total completion time are rarely considered. In addition, the focus lays on energy consumption, while fluctuating energy prices are less examined so far.

These findings lead to the second research question **Q2**.[3] Here, the dependencies of total tardiness and electricity costs are analysed, taking into account time-of-use tariffs and variable production speeds. In principle, a time-indexed and a sequence-based formulation seems suitable for MIP modelling. While the time-indexed formulation requires considerably more constraints as well as continuous variables, the number of binary variables increases more with the problem size in the sequence-based formulation. A main reason lays in the calculation of the energy costs of the time-indexed formulation, which is done during the run-time. The sequence-based formulation calculates all scenarios in advance and then selects them by binary variables. Overall, this outsourcing of cost calculation seems to be favourable, which is why the sequence-dependent formulation is recommended. This conclusion is not only based on the problem size complexity but also on computational complexity. Both models do not differ significantly and are only suitable for solving small problem sizes.

The interdependencies between both objectives can be examined on small instances using CPLEX 12.6. The acceptance of small additional delays allows significant reductions in electricity costs. Energy costs can be reduced by 3.8 % if total tardiness is increased by 1.6 %. 16 % more delays lead in average to 21.9 % lower energy costs. Thus, the saving of energy costs increases degressively with higher total tardiness. Thereby, speed reductions have a greater impact on savings than the exploitation of TOU price fluctuations. From

[3] Q2: To what extent can electricity costs be reduced by changing production speeds and deliberate delays and how can that problem be mathematically formalized?

this, it can be concluded that a reduction in consumption is more beneficial than shifting loads to off-peak times. Interestingly, however, a reduction in energy costs is not always accompanied by a reduction in consumption. Some pareto optimal solutions show lower costs at higher electricity consumption.

Similar results are observed in Chapter 4. Here, the focus lays on research question **Q3**.[4] The basic idea is to develop a hybrid meta-heuristic approach that combines the strengths of diversification of population-based PSO with the intensification of tabu search. The encoding of the solutions is done by means of two arrays separated from each other. On the one hand, the speed of each job is defined at each production stage. On the other hand, sequence and machine allocation at each stage are represented in the second array. Non-active schedules are considered within the decoding. PXO is used for crossovers in the PSO. For intensification in the tabu search, job sequence and machine assignment is influenced by swap and insertion moves. Speed changes are made job-specific.

To evaluate the HPSO, the results are compared with both, exact solutions from the model and other heuristics. The HPSO reproduces the optimal pareto front for smaller instances very well. When comparing with NSGA-II and different PSO variants, runtime behavior and solution quality proves to be advantageous. A total of five comparison criteria is used for evaluation. The differences are also proven to be significant by means of two-sample t-test. As a general recommendation can be summarised: 1) Within the solution archive a good diversification should be achieved, for which population-based metaheuristics like PSO seem suitable. 2) For intensification, a tabu search is much more effective than local search or simulated annealing. 3) With regard to de-/encoding, a separation of intensity and machine allocation works well.

The second considered HFS problems serves to analyse the relationship between minimizing peak power and exploiting fluctuating electricity prices. At the same time, production capacities should be used efficiently, which results in **Q4**.[5] First, a single objective MIP was established in Chapter 5 to answer this question. By parametrically optimizing the peak power value, it could be determined that a levelling of power consumption initially has only a minor impact on the overall costs. A reduction of 20 % in peak power increases electricity costs based on the consumption charge by 6 % with slightly lower total energy consumption. In any case, the consideration of energy costs is favourable. For example, electricity costs were 5 % higher (and energy consumption 14 %) if electricity was not included in the objective function.

[4]Q3: Which heuristic is suitable to solve a hybrid flow shop scheduling problem with variable execution modes and total tardiness as well as energy costs as objectives?

[5]Q4: How do capacitive scheduling criteria interact with peak power and energy costs as additional objective functions?

The minimization of peak power and the exploitation of fluctuating electricity prices are contradictory. This was also evident in the follow-up study in Chapter 6. Here the model is transformed into a three objective MIP. Namely makespan, peak power and total electricity costs are minimized simultaneously. Degressive progression can also be identified in the three-dimensional pareto front. Already small extensions of the maximum completion time lead to significant reductions in energy costs or peak power. To solve larger problem instances an ILS with problem-specific characteristics is proposed, which also serves to answer research question **Q5**.[6] Pareto optimal solutions fulfil different purposes and are therefore often very different in their composition. In order to efficiently search the entire solution space often population based heuristics are used. But also single solution search methods can be suitable if the solution archive is well managed and solutions are regularly perturbed in such a way that new areas of the solution space are reached without destroying the "good" properties of the current solution. In the proposed ILS a block move operator proved to be efficient for the perturbation.

Solutions are represented in the ILS by a first stage job sequence encoding. The exact scheduling on further stages including the machine allocation is done by three different list scheduling approaches. While ECT (earliest completion time) is used to find good solutions for the makespan criterion, with DMS (deterministic machine selection) and LMS (levelling machine selection) two new algorithms have been developed which are sufficient for the energy-oriented objectives. The active plans generated in this way are checked for energy cost savings using a right-shifting algorithm. Overall, the ILS can efficiently solve the problem and shows a similar shape as the optimal three dimensional pareto front with a much shorter computation time. In the computational study, based on 13500 test runs, the ILS outperforms significantly the NSGA-II. The significance was again proven by means of a t-test for all three comparison criteria computing time, hypervolume and number of NDS. Thus, **Q5** can be confirmed.

Finally, question **Q6**[7] focuses on the impact of subcontracting options on EAS. While the field of EAS is intensively discussed in the literature, only few contributions exist on scheduling with subcontracting options. Nevertheless, the existing contributions show that costs can be significantly reduced by subcontracting certain processing steps. This could also be confirmed in initial tests with regard to energy costs. Especially in energy-intensive industries such as paper production, metal processing or chemical industry, energy costs can account for more than half of the variable gross production costs. If energy prices increase significantly, outsourcing can be reasonable. This applies, among other things, to

[6] Q5: Is an iterated local search algorithm suitable to find pareto optimal solutions in a three-objective energy aware hybrid flow shop problem?

[7] Q6: Which influence has subcontracting on energy aware scheduling?

scenarios in which external companies take over processing operations at fixed prices or where plants of the same company have access to more favourable electricity prices, for example through own power plants or better contracts.

In detail, Chapter 7 provides a first MIP formulation for the problem. As objective serves total cost for internal and external production as well as transport expenses due to subcontracting. RTP from the stock exchange (EEX) are used for the calculation of the energy costs. Chapter 8 proposes a GA to solve the problem for larger instances. To reflect the subcontracting decisions for each solution, a complex matrix encoding was chosen. In order to eliminate inefficiencies in this complex coding, an algorithm for intelligent swaps was integrated into the solution evaluation, which uses idle times on the machines to schedule available jobs earlier. In addition, a right-shifting procedure serves to exploit the fluctuation of electricity prices. In over 2500 runs, the GA deviates from the optimal solution for 84 different problem sizes by only 0.36 %. Thereby, the heuristic requires less than 0.2 % of the solver's computing time (CPLEX 12.6).

Overall, the relocation of production steps appears to be advantageous in the first test scenarios. In the instances under consideration, the costs could be reduced by up to 7 % due to intelligent subcontracts. It was assumed that external production costs are on average 20 % higher than the most expensive internal machines. Transport costs also increase expenditure. Especially in times of very high electricity prices or high machine utilization, subcontracting pays off.

9.2 Critical review and future research

Each model represents a simplification of reality and thus contains inadequacies. The methodical approach of this work is described in section 1.4 and shall enable valid answers to the research questions. Nevertheless, there are some problem characteristics in reality which have not been represented so far. For example, neither dynamic nor stochastic influences that occur in daily operations in industrial production were taken into account. Unfortunately, such factors are difficult to predict and bring therefore new uncertainties into the planning. Within the three HFS problems considered, some problem specific simplifications have been made. These should not remain unmentioned at this point:

HFS1: In the first problem discrete execution modes are analysed for variable speed. However, electric motors, which represent the majority of industrial consumers, can be continuously adjusted in speed. Consequently, a further development towards continuous speed functions would be of interest. This would possibly provide additional savings opportunities, but the general relationships are not expected to change.

HFS2: With regard to peak power optimization, the general suitability of scheduling must be questioned. While the peak power is charged for long-term billing periods such as months or a year, operational control is usually on a daily or weekly basis. From this can be derived that within scheduling the peak demand is ideally modelled as a soft constraint similar to classical energy management systems. The default value could be taken from historical data or forecasts. An overshoot could be allowed by penalty costs within the objective function(s).

HFS3: Some assumptions can also be questioned with regard to subcontracting decisions. It is assumed that subcontractors are available at all times. In reality, however, this is not the case for every processing step. Also, certain processes may be key competencies of the company and cannot be outsourced for compliance reasons. Such restrictions could be integrated into the problem formulation.

All these points certainly serve to bring the models closer to reality. Nevertheless, the data used and assumptions made in this dissertation are taken from the literature when no real data are available and are verified to the very best knowledge. It should be the task of future research to apply the new methods presented to different real life problems. In this context, uncertainties should possibly also be taken into account. This applies in particular to the development of the electricity price. In the EAS literature, often historical electricity prices are used. In reality, however, forecasted prices are needed. Research that combines both is lacking and represents a promising further development of current EAS approaches.

Another point that should be investigated in the future and which generally receives rather little attention in multi-criteria EAS, is the selection of a single solution. The presented approaches deal with the determination of all pareto optimal solutions. The choice of a suitable solution depends strongly on the practical application and is linked to the respective company. However, there exist various scientific methods in the area of multi-criteria decision making.[8] One of the most commonly used approaches is Fuzzy TOPSIS. The idea is that the best solution should be as close as possible to the theoretically best solution and as far away as possible from the anti-optimal point.[9] Other known approaches are for example the analytical hierarchy process, weighted sum or the best worst method. The selection of a suitable method is in itself an optimization problem which is often called a paradox.

[8] An overview gives for example Triantaphyllou (2000): *Multi-Criteria Decision Making Methods.*
[9] Cf. Lai / Liu / Hwang (1994): *Topsis for MODM.*

Irrespective of how the decision-maker selects a solution, this work has shown that the consideration of energy costs can lead to enormous cost savings. The consideration of time-dependent electricity prices alone can reduce electricity costs by 20 %.[10] In addition, significant reductions in consumption can be achieved without significantly affecting makespan or tardiness. The heuristic algorithms developed in this thesis are able to solve complex real problems. The application can ultimately make a significant contribution to sustainability in industrial production in the future.

[10]This value corresponds to Castro / Harjunkoski / Grossmann (2009): *New Continuous-Time Scheduling.*

A Declarations of authorship

I hereby certify that I have authored this Dissertation to achieve the academic degree Doctor rerum politicarum (Dr. rer. pol.) entitled

"Energy aware hybrid flow shop scheduling"

independently and without undue assistance from third parties. No other than the resources and references indicated in this thesis have been used. I have marked both literal and accordingly adopted quotations as such. The own share of publications which were created in joint work is declared in the following. There were no additional persons involved in the intellectual preparation of the present thesis. I am aware that violations of this declaration may lead to subsequent withdrawal of the degree.

Dresden, 30th April 2020

[redacted]

Sven Schulz

TECHNISCHE UNIVERSITÄT DRESDEN

Fakultät Wirtschaftswissenschaften Lehrstuhl für BWL, insb. Industrielles Management, Prof. Dr. Udo Buscher

Declaration of authorship of a publication

Title	Multi-objective Hybrid Flow Shop Scheduling with Variable Discrete Production Speed Levels and Time-Of-Use Energy Prices
In	Manuscript submitted to *Journal of Business Economics* Current status: accepted

The aforementioned publication is written in collaboration by the following authors:

Author 1:	Sven Schulz (SS)
Author 2:	Udo Buscher (UB)
Author 3:	Liji Shen (LS)

The individual contribution of each participating author is stated in the table hereinafter. Underlined initials, if any, indicate authors with a major share in the respective field.

Establishing of research topic	SS
Literature review	SS
Conducting of research	SS
Appraisal and interpretation of results	SS
Publication strategy	SS, UB
Writing of manuscript	SS
Pre-review and lectorship	SS, UB, LJ

By their signature, the authors declare their agreement with the above statement.

Dresden, 10 April 2020

[redacted]

Postadresse (Briefe)
TU Dresden, 01062 Dresden

Postadresse (Pakete u.ä.)
TU Dresden
Helmholtzstraße 10
01069 Dresden

Besucheradresse:
Helmholtzstr. 10
Hülsse-Bau
Zimmer N 503

Zufahrt
Seiteneingang, gekennzeichnete Parkfläche im Innenhof

Internet
http://tu-dresden.de

TECHNISCHE UNIVERSITÄT DRESDEN

Fakultät Wirtschaftswissenschaften Lehrstuhl für BWL, insb. Industrielles Management, Prof. Dr. Udo Buscher

Declaration of authorship of a publication

Title	Energy Aware Scheduling in Flexible Flow Shops with Hybrid Particle Swarm Optimization
In	Manuscript submitted to *Computers & Operations Research* Current status: major revision

The aforementioned publication is written in collaboration by the following authors:

Author 1:	Junwen Ding (JD)
Author 2:	Sven Schulz (SS)
Author 3:	Liji Shen (LS)
Author 4:	Udo Buscher (UB)
Author 5:	Zhipeng Lü (ZL)

The individual contribution of each participating author is stated in the table hereinafter. Underlined initials, if any, indicate authors with a major share in the respective field.

Establishing of research topic	SS
Literature review	SS
Conducting of research	JD, SS, LS
Appraisal and interpretation of results	JD, SS, LS
Publication strategy	LS, UB, ZL
Writing of manuscript	SS, LS
Pre-review and lectorship	SS, LS, UB, ZL

By their signature, the authors declare their agreement with the above statement.

Dresden, 10 April 2020

[redacted]

Postadresse (Briefe)
TU Dresden, 01062 Dresden

Postadresse (Pakete u.ä.)
TU Dresden
Helmholtzstraße 10
01069 Dresden

Besucheradresse:
Helmholtzstr. 10
Hülsse-Bau
Zimmer N 503

Zufahrt
Seiteneingang, gekennzeichnete Parkfläche im Innenhof

Internet
http://tu-dresden.de

TECHNISCHE UNIVERSITÄT DRESDEN

Fakultät Wirtschaftswissenschaften Lehrstuhl für BWL, insb. Industrielles Management, Prof. Dr. Udo Buscher

Declaration of authorship of a publication

Title	A multi-objective iterated local search algorithm for comprehensive energy-aware hybrid flow shop scheduling
In	*Journal of Cleaner Production* 224 (2019), pp. 421-434.

The aforementioned publication is written in collaboration by the following authors:

Author 1:	Sven Schulz (SS)
Author 2:	Janis S. Neufeld (JN)
Author 3:	Udo Buscher (UB)

The individual contribution of each participating author is stated in the table hereinafter. Underlined initials, if any, indicate authors with a major share in the respective field.

Establishing of research topic	SS
Literature review	SS, JN
Conducting of research	SS
Appraisal and interpretation of results	SS
Publication strategy	SS, UB
Writing of manuscript	SS, JN
Pre-review and lectorship	SS, JN, UB

By their signature, the authors declare their agreement with the above statement.

Dresden, 10 April 2020

[redacted]

Postadresse (Briefe)
TU Dresden, 01062 Dresden

Postadresse (Pakete u.ä.)
TU Dresden
Helmholtzstraße 10
01069 Dresden

Besucheradresse:
Helmholtzstr. 10
Hülsse-Bau
Zimmer N 503

Zufahrt
Seiteneingang, gekennzeichnete Parkfläche im Innenhof

Internet
http://tu-dresden.de

TECHNISCHE UNIVERSITÄT DRESDEN

Fakultät Wirtschaftswissenschaften Lehrstuhl für BWL, insb. Industrielles Management, Prof. Dr. Udo Buscher

Declaration of authorship of a publication

Title	Hybrid Flow Shop Scheduling with Subcontracting Options and Time-Depending Energy Costs
In	*Logistikmanagement - Beiträge zur LM 2017,* pp. 163–170, *Stuttgart*.

The aforementioned publication is written in collaboration by the following authors:

Author 1:	Sven Schulz (SS)
Author 2:	Sophie Apelmeier (SA)
Author 3:	Udo Buscher (UB)

The individual contribution of each participating author is stated in the table hereinafter. Underlined initials, if any, indicate authors with a major share in the respective field.

Establishing of research topic	SS
Literature review	SA
Conducting of research	SS, SA
Appraisal and interpretation of results	SS, SA
Publication strategy	SS
Writing of manuscript	SS
Pre-review and lectorship	SS, SA, UB

By their signature, the authors declare their agreement with the above statement.

Dresden, 03 April 2020

[redacted]

Postadresse (Briefe)
TU Dresden, 01062 Dresden

Postadresse (Pakete u.ä.)
TU Dresden
Helmholtzstraße 10
01069 Dresden

Besucheradresse:
Helmholtzstr. 10
Hülsse-Bau
Zimmer N 503

Zufahrt
Seiteneingang, gekennzeichnete Parkfläche im Innenhof

Internet
http://tu-dresden.de

Bibliography

Abdelaziz, E. / Saidur, R. / Mekhilef, S. (2011): A review on energy saving strategies in industrial sector. In: *Renewable and Sustainable Energy Reviews*, vol. 15, no. 1, pp. 150–168.

Almeida, A. T. de / Ferreira, F. J. / Both, D. (2005): Technical and economical considerations in the application of variable-speed drives with electric motor systems. In: *IEEE Transactions on Industry Applications*, vol. 41, no. 1, pp. 188–199.

Anke, C.-P. / Dierstein, C. / Hladik, D. / Möst, D. (2018): Begleitstudie WindNODE – Lastverschiebepotentiale in Dresden. In: *Schriften des Lehrstuhls für Energiewirtschaft, TU Dresden*. Volume 16. Technische Universität Dresden.

Arora, S. / Barak, B. (2009): Computational complexity: a modern approach. Computational complexity: Cambridge University Press.

Artigues, C. / Lopez, P. / Hait, A. (2013): The energy scheduling problem: Industrial case-study and constraint propagation techniques. In: *International Journal of Production Economics*, vol. 143, no. 1, pp. 13–23.

Ashok, S. (2006): Peak-load management in steel plants. In: *Applied Energy*, vol. 83, no. 5, pp. 413–424.

Babu, C. / Ashok, S. (2008): Peak Load Management in Electrolytic Process Industries. In: *IEEE Transactions on Power Systems*, vol. 23, no. 2, pp. 399–405.

Bathurst, G. N. / Weatherill, J. / Strbac, G. (2002): Trading wind generation in short term energy markets. In: *IEEE Transactions on Power Systems*, vol. 17, no. 3, pp. 782–789.

Beale, E. M. L. / Forrest, J. J. H. (1976): Global optimization using special ordered sets. In: *Mathematical Programming*, vol. 10, no. 1, pp. 52–69.

Bego, A. / Li, L. / Sun, Z. (2014): Identification of reservation capacity in critical peak pricing electricity demand response program for sustainable manufacturing systems. In: *International Journal of Energy Research*, vol. 38, no. 6, pp. 728–736.

Benavides, A. J. / Ritt, M. / Miralles, C. (2014): Flow shop scheduling with heterogeneous workers. In: *European Journal of Operational Research*, vol. 237, no. 2, pp. 713–720.

Bertrand, J. W. M. / Fransoo, J. C. (2002): Operations management research methodologies using quantitative modeling. In: *International Journal of Operations & Production Management*, vol. 22, no. 2, pp. 241–264.

Besikci, U. / Bilge, Ü. / Ulusoy, G. (2015): Multi-mode resource constrained multi-project scheduling and resource portfolio problem. In: *European Journal of Operational Research*, vol. 240, no. 1, pp. 22–31.

Beume, N. / Naujoks, B. / Emmerich, M. (2007): SMS-EMOA: Multiobjective selection based on dominated hypervolume. In: *European Journal of Operational Research*, vol. 181, no. 3, pp. 1653–1669.

Biel, K. / Glock, C. H. (2016): Systematic literature review of decision support models for energy-efficient production planning. In: *Computers & Industrial Engineering*, vol. 101, pp. 243–259.

BMWI (2019): Gesamtausgabe der Energiedaten - Datensammlung des BMWi. Visited on 02.04.2020. url: https://www.bmwi.de/Redaktion/DE/Binaer/Energiedaten/energiedaten-gesamt-xls.xlsx?__blob=publicationFile&v=117.

Boukas, E.-K. / Haurie, A. / Soumis, F. (1990): Hierarchical approach to steel production scheduling under a global energy constraint. In: *Annals of Operations Research*, vol. 26, pp. 289–311.

BP Energy Economics (2019): BP Energy Outlook 2019. Technical report. London, United Kingdom.

Brah, S. A. / Loo, L. L. (1999): Heuristics for scheduling in a flow shop with multiple processors. In: *European Journal of Operational Research*, vol. 113, no. 1, pp. 113–122.

Branke, J. / Deb, K. / Miettinen, K. / Roman, S. (2008): Multiobjective Optimization, Interactive and Evolutionary Approaches [outcome of Dagstuhl seminars]. Berlin Heidelberg: Springer.

Brucker, P. / Knust, S. (2009): Complexity results for scheduling problems. Ed. by U. Osnabrück. Visited on 31.03.2020. url: `http://www2.informatik.uni-osnabrueck.de/knust/class/`.

Bruzzone, A. / Anghinolfi, D. / Paolucci, M. / Tonelli, F. (2012): Energy-aware scheduling for improving manufacturing process sustainability: A mathematical model for flexible flow shops. In: *CIRP Annals - Manufacturing Technology*, vol. 61, no. 1, pp. 459–462.

Bundesamt für Justiz (2006): Konzessionsabgabenverordnung §2 (7).

Castro, P. / Harjunkoski, I. / Grossmann, I. (2009): New Continuous-Time Scheduling Formulation for Continuous Plants under Variable Electricity Cost. In: *Industrial & Engineering Chemistry Research*, vol. 48, no. 14, pp. 6701–6714.

Castro, P. / Harjunkoski, I. / Grossmann, I. (2011): Optimal scheduling of continuous plants with energy constraints. In: *Computers & Chemical Engineering*, vol. 35, no. 2, pp. 372–387.

Che, A. / Wu, X. / Peng, J. / Yan, P. (2017): Energy-efficient bi-objective single-machine scheduling with power-down mechanism. In: *Comput Oper Res*, vol. 85, pp. 172–183.

Che, A. / Zhang, S. / Wu, X. (2017): Energy-conscious unrelated parallel machine scheduling under time-of-use electricity tariffs. In: *J Clean Prod*, vol. 156, pp. 688–697.

Chen, B. / Zhang, X. (2019): Scheduling with time-of-use costs. In: *European Journal of Operational Research*, vol. 274, no. 3, pp. 900–908.

Chen, Z.-L. / Li, C.-L. (2008): Scheduling with subcontracting options. In: *IIE Transactions*, vol. 40, no. 12, pp. 1171–1184.

Cheng, T. / Guoqing Wang (1999): A note on scheduling the two-machine flexible flowshop. In: *IEEE Transactions on Robotics and Automation*, vol. 15, no. 1, pp. 187–190.

Chircop, K. / Zammit-Mangion, D. (2013): On Epsilon-Constraint Based Methods for the Generation of Pareto Frontiers. In: *Journal of Mechanics Engineering and Automation*, vol. 3, pp. 279–289.

Choi, B.-C. / Chung, K. (2016): Min–max regret version of a scheduling problem with outsourcing decisions under processing time uncertainty. In: *European Journal of Operational Research*, vol. 252, no. 2, pp. 367–375.

Choi, S.-W. / Kim, Y.-D. / Lee, G.-C. (2005): Minimizing total tardiness of orders with reentrant lots in a hybrid flowshop. In: *International Journal of Production Research*, vol. 43, no. 11, pp. 2149–2167.

Ciro, G. C. / Dugardin, F. / Yalaoui, F. / Kelly, R. (2016): A NSGA-II and NSGA-III comparison for solving an open shop scheduling problem with resource constraints. In: *IFAC-PapersOnLine*, vol. 49, no. 12. 8th IFAC Conference on Manufacturing Modelling, Management and Control MIM 2016, pp. 1272–1277.

Dai, M. / Tang, D. / Giret, A. / Salido, M. / Li, W. (2013): Energy-efficient scheduling for a flexible flow shop using an improved genetic-simulated annealing algorithm. In: *Robotics and Computer-Integrated Manufacturing*, vol. 29, no. 5, pp. 418–429.

Dai, M. / Tang, D. / Xu, Y. / Li, W. (2015): Energy-aware integrated process planning and scheduling for job shops. In: *Journal of Energineering Manufacture*, vol. 229, no. 1, pp. 13–26.

Deb, K. / Pratap, A. / Agarwal, S. / Meyarivan, T. (2002): A fast and elitist multiobjective genetic algorithm: NSGA-II. In: *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197.

Deb, K. (2014): Multi-objective optimization. In: *Search methodologies*. Ed. by E. K. Burke / G. Kendall. Springer, pp. 403–449.

Deb, K. / Jain, H. (2012): Handling many-objective problems using an improved NSGA-II procedure. In: *Evolutionary computation (CEC), 2012 IEEE congress on.* IEEE, pp. 1–8.

Dessouky, M. M. / Dessouky, M. I. / Verma, S. K. (1998): Flowshop scheduling with identical jobs and uniform parallel machines. In: *European Journal of Operational Research*, vol. 109, no. 3, pp. 620–631.

Ding, J.-Y. / Song, S. / Zhang, R. / Chiong, R. / Wu, C. (2016): Parallel machine scheduling under time-of-use electricity prices: New models and optimization approaches. In: *IEEE Transactions on Automation Science and Engineering*, vol. 13, no. 2, pp. 1138–1154.

Ding, J. / Schulz, S. / Shen, L. / Buscher, U. / Lü, Z. (2020): Energy Aware Scheduling in Flexible Flow Shops with Hybrid Particle Swarm Optimization. In: *Computers & Operations Research*, Submitted Manuscript - Second Revision.

Dolgui, A. / Kovalev, S. / Pesch, E. (2015): Approximate solution of a profit maximization constrained virtual business planning problem. In: *Omega*, vol. 57, no. PB, pp. 212–216.

Domschke, W. / Scholl, A. (2008): Grundlagen der Betriebswirtschaftslehre - Eine Einführung aus entscheidungsorientierter Sicht. 4th edition. Berlin, Heidelberg: Springer-Verlag.

Dong, M. (2013): Parallel machine scheduling with limited controllable machine availability. In: *International Journal of Production Research*, vol. 51, no. 8, pp. 2240–2252.

Du, B. / Chen, H. / Luo, H. / Li, X. / Huang, G. Q. (2013): Hybrid flow shop scheduling considering machine electricity consumption cost. In: *International Journal of Production Economics*, vol. 146, no. 2, pp. 423–439.

Du, J. / Leung, J. Y.-T. (1990): Minimizing total tardiness on one machine is NP-hard. In: *Mathematics of operations research*, vol. 15, no. 3, pp. 483–495.

Dubois-Lacoste, J. / Lopez-Ibanez, M. / Stützle, T. (2011): A hybrid TP+ PLS algorithm for bi-objective flow-shop scheduling problems. In: *Computers & Operations Research*, vol. 38, no. 8, pp. 1219–1236.

Ebrahimi, M. / Ghomi, S. F. / Karimi, B. (2014): Hybrid flow shop scheduling with sequence dependent family setup time and uncertain due dates. In: *Applied Mathematical Modelling*, vol. 38, no. 9, pp. 2490–2504.

EEX (2020): Phelix Spot Market Prices. Ed. by E. E. E. AG. Visited on 01.04.2020. url: https://www.eex.com/en/market-data/power/power-indices/.

Eiselt, H. A. / Sandblom, C.-L. (2004): Decision analysis, location models, and scheduling problems. Berlin Heidelber: Springer Science & Business Media.

Emmerich, M. T. / Deutz, A. H. (2018): A tutorial on multiobjective optimization: fundamentals and evolutionary methods. In: *Natural computing*, vol. 17, no. 3, pp. 585–609.

Enerdata (2019): Global Energy Statistical Yearbook 2019. Visited on 12.03.2020. url: https://yearbook.enerdata.net/.

Fang, K. / Uhan, N. / Zhao, F. / J. W., S. (2011): A new approach to scheduling in manufacturing for power consumption and carbon footprint reduction. In: *Journal of Manufacturing Systems*, vol. 30, no. 4, pp. 234–240.

Fang, K. / Uhan, N. / Zhao, F. / Sutherland, J. (2013): Flow shop scheduling with peak power consumption constraints. In: *Annals of Operations Research*, vol. 206, no. 1, pp. 115–145.

Fang, K. / Lin, M. (2013): Parallel-machine scheduling to minimize tardiness penalty and power cost. In: *Computers and Industrial Engineering*, vol. 64, no. 1, pp. 224–234.

Framinan, J. M. / Leisten, R. (2008): A multi-objective iterated greedy search for flowshop scheduling with makespan and flowtime criteria. In: *OR Spectrum*, vol. 30, no. 4, pp. 787–804.

Fraunhofer, I. S. I. (2015): Electricity Costs of Energy Intensive Industries. In: *An International Comparison*, Gahm, C. / Denz, F. / Dirr, M. / Tuma, A. (2016): Energy-efficient scheduling in manufacturing companies: A review and research framework. In: *Eur J Oper Res*, vol. 248, no. 3, pp. 744–757.

Geiger, M. J. (2011): Decision support for multi-objective flow shop scheduling by the Pareto iterated local search methodology. In: *Computers & Industrial Engineering*, vol. 61, no. 3, pp. 805–812.

Gerstl, E. / Mosheiov, G. (2014): The optimal number of used machines in a two-stage flexible flowshop scheduling problem. In: *Journal of Scheduling*, vol. 17, no. 2, pp. 199–210.

Goldberg, D. E. / Richardson, J. (1987): Genetic algorithms with sharing for multimodal function optimization. In: *The Second International Conference on Genetic Algorithms*. Lawrence Erlbaum Associates, pp. 41–49.

Gonzalez-Neira, E. M. / Garcia-Caceres, R. G. / Caballero-Villalobos, J. P. / Molina-Sanchez, L. P. / Montoya-Torres, J. R. (2016): Stochastic flexible flow shop scheduling problem under quantitative and qualitative decision criteria. In: *Computers & Industrial Engineering*, vol. 101, pp. 128–144.

Graham, R. / Lawler, E. / Lenstra, J. / Rinnooy Kan, A. (1979): Optimization and approximation in deterministic sequencing and scheduling: A survey. In: *Annals of Discrete Mathematics*, vol. 5, no. 1, pp. 287–326.

Gupta, J. / Hariri, A. / Potts, C. (1997): Scheduling a two-stage hybrid flow shop with parallel machines at the first stage. In: *Annals of Operations Research*, vol. 69, no. 0, pp. 171–191.

Györgyi, P. / Kis, T. (2018): Minimizing the maximum lateness on a single machine with raw material constraints by branch-and-cut. In: *Computers & Industrial Engineering*, vol. 115, pp. 220–225.

Hagedorn, G. / Kalmus, P. / Mann, M. / Vicca, S. / Van den Berge, J. / Ypersele, J.-P. van / Bourg, D. / Rotmans, J. / Kaaronen, R. / Rahmstorf, S. / Kromp-Kolb, H. / Kirchengast, G. / Knutti, R. / Seneviratne, S. I. / Thalmann, P. / Cretney, R. / Green, A. / Anderson, K. / Hedberg, M. / Nilsson, D. /

KUTTNER, A. / HAYHOE, K. (2019): Concerns of young protesters are justified. In: *Science*, vol. 364, no. 6436. Ed. by J. SILLS, pp. 139–140.

HAHN, G. J. / SENS, T. / DECOUTTERE, C. / VANDAELE, N. J. (2016): A multi-criteria approach to robust outsourcing decision-making in stochastic manufacturing systems. In: *Computers & Industrial Engineering*, vol. 98, pp. 275–288.

HAIT, A. / ARTIGUES, C. (2011a): An hybrid CP/MILP method for scheduling with energy costs. In: *European Journal of Industrial Engineering*, vol. 5, no. 4, pp. 471–489.

HAIT, A. / ARTIGUES, C. (2011b): On electrical load tracking scheduling for a steel plant. In: *Computers & Chemical Engineering*, vol. 35, no. 12, pp. 3044–3047.

HONG, J. / LEE, J. (2016): Outsourcing decisions in single machine scheduling problem with multiple external facilities, in: Journal of Marine Science and Technology. In: *Journal of Marine Science and Technology*, vol. 24, no. 3, pp. 603–609.

HUANG, R. H. / YU, T. H. / YANG, C. L. (2013): Multiprocessor Flow Shop Scheduling Problem with Common due Window. In: *Innovation for Applied Science and Technology.* Volume 284. Applied Mechanics and Materials. Trans Tech Publications Ltd, pp. 3712–3716.

IEA (2007): Tracking Industrial Energy Efficiency and CO2 Emissions. Report. International Energy Agency, p. 324.

IEA (2018): World Energy Balances: Overview. In: *International Energy Agency, Paris, France*,

JASZKIEWICZ, A. (2018): Many-Objective Pareto Local Search. In: *European Journal of Operational Research*, vol. in press.

JI, M. / WANG, J. / LEE, W. (2013): Minimizing resource consumption on uniform parallel machines with a bound on makespan. In: *Computers & operations research*, vol. 40, no. 12, pp. 2970–2974.

JIANG, K. / HE, C. / XU, X. / JIANG, W. / XIANG, P. / LI, H. / LIU, J. (2018): Transition scenarios of power generation in China under global 2°C and 1.5°C targets. In: *Global Energy Interconnection*, vol. 1, no. 4, pp. 477–486.

Jiang, S.-L. / Zhang, L. (2019): Energy-oriented scheduling for hybrid flow shop with limited buffers through efficient multi-objective optimization. In: *IEEE Access*,

Johnson, S. M. (1954): Optimal two- and three-stage production schedules with setup times included. In: *Naval Research Logistics Quarterly*, vol. 1, no. 1, pp. 61–68.

Jungwattanakit, J. / Reodecha, M. / Chaovalitwongse, P. / Werner, F. (2008): Algorithms for flexible flow shop problems with unrelated parallel machines, setup times, and dual criteria. In: *The International Journal of Advanced Manufacturing Technology*, vol. 37, no. 3, pp. 354–370.

Karimi, N. / Zandieh, M. / Karamooz, H. (2010): Bi-objective group scheduling in hybrid flexible flowshop: A multi-phase approach. In: *Expert Systems with Applications*, vol. 37, no. 6, pp. 4024–4032.

Kaya, D. / Yagmur, E. / Yigit, K. / Kilic, F. / Eren, A. / Celik, C. (2008): Energy efficiency in pumps. In: *Energy Conversion and Management*, vol. 49, no. 6, pp. 1662–1673.

Keha, A. B. / Khowala, K. / Fowler, J. W. (2009): Mixed integer programming formulations for single machine scheduling problems. In: *Computers & Industrial Engineering*, vol. 56, no. 1, pp. 357–367.

Kennedy, J. / Eberhart, R. (1995): Particle swarm optimization. In: *Proceedings of 1995 IEEE International Conference on Neural Networks*. Volume 4. IEEE, pp. 1942–1948. isbn: 0-7803-2768-3.

Kennedy, J. / Eberhart, R. (1997): A discrete binary version of the particle swarm algorithm. In: *IEEE International Conference on Systems, Man, and Cybernetics. Computational Cybernetics and Simulation*. Volume 5. IEEE, pp. 4104–4108. isbn: 0-7803-4053-1.

Kirshenblatt-Gimblett, B. (2006): Part 1, what is research design. In: *The Context of Design." Performance Studies Methods Course syllabus. New York University, Spring*,

Kis, T. / Pesch, E. (2005): A review of exact solution methods for the non-preemptive multiprocessor flowshop problem. In: *European Journal of Operational Research*, vol. 164, no. 3, pp. 592–608.

Küster, T. / Lützenberger, M. / Freund, D. / Albayrak, S. (2013): Distributed evolutionary optimisation for electricity price responsive manufacturing using multi-agent system technology. In: *Int Journal On Advances in Intelligent Systems*, vol. 7.

Lai, Y.-J. / Liu, T.-Y. / Hwang, C.-L. (1994): Topsis for MODM. In: *European journal of operational research*, vol. 76, no. 3, pp. 486–500.

Laurijssen, J. / Faaij, A. / Worrell, E. (2013): Benchmarking energy use in the paper industry: a benchmarking study on process unit level. In: *Energy Efficiency*, vol. 6, no. 1, pp. 49–63.

Le, C. / Pang, C. (2013): Fast reactive scheduling to minimize tardiness penalty and energy cost under power consumption uncertainties. In: *Computers and Industrial Engineering*, vol. 66, no. 2, pp. 406–417.

Lee, C. (2018): A review of applications of genetic algorithms in operations management. In: *Engineering Applications of Artificial Intelligence*, vol. 76, pp. 1–12.

Lee, I. S. / Sung, C. S. (2008): Single machine scheduling with outsourcing allowed. In: *International Journal of Production Economics*, vol. 111, pp. 623–634.

Lee, K. / Choi, B.-C. (2011): Two-stage production scheduling with an outsourcing option. In: *European Journal of Operational Research*, vol. 213, pp. 489–497.

Lee, T. / Loong, Y. (2019): A review of scheduling problem and resolution methods in flexible flow shop. In: *Journal of Industrial Engineering Computations*, vol. 10, no. 1, pp. 67–88.

Lei, D. / Gao, L. / Zheng, Y. (2018): A novel teaching-learning-based optimization algorithm for energy-efficient scheduling in hybrid flow shop. In: *IEEE Transactions on Engineering Management*, vol. 65, no. 2, pp. 330–340.

Lei, D. / Guo, X. (2016): Hybrid flow shop scheduling with not-all-machines options via local search with controlled deterioration. In: *Computers & Operations Research*, vol. 65, pp. 76–82.

LEI, D. / ZHENG, Y. / GUO, X. (2017): A shuffled frog-leaping algorithm for flexible job shop scheduling with the consideration of energy consumption. In: *International Journal of Production Research*, vol. 55, no. 11, pp. 3126–3140.

LI, D. / MENG, X. / LIANG, Q. / ZHAO, J. (2015): A heuristic-search genetic algorithm for multi-stage hybrid flow shop scheduling with single processing machines and batch processing machines. In: *Journal of Intelligent Manufacturing*, vol. 26, no. 5, pp. 873–890.

LI, J.-q. / SANG, H.-y. / HAN, Y.-y. / WANG, C.-g. / GAO, K.-z. (2018): Efficient multi-objective optimization algorithm for hybrid flow shop scheduling problems with setup energy consumptions. In: *Journal of Cleaner Production*, vol. 181, pp. 584–598.

LI, L. / LUAN, J. / QIU, Y. (2016): Two-Stage Flowshop Scheduling with Outsourcing Allowed. In: *International Journal of u- and e-Service, Science and Technology*, vol. 9, no. 10, pp. 245–254.

LI, M. / LEI, D. / CAI, J. (2019): Two-level imperialist competitive algorithm for energy-efficient hybrid flow shop scheduling problem with relative importance of objectives. In: *Swarm and Evolutionary Computation*, vol. 49, pp. 34–43.

LINN, R. / ZHANG, W. (1999): Hybrid flow shop scheduling: A survey. In: *Computers & Industrial Engineering*, vol. 37, no. 1-2, pp. 57–61.

LIU, C. / ZHAO, C. / XU, Q. (2012): Integration of electroplating process design and operation for simultaneous productivity maximization, energy saving, and freshwater minimization. In: *Chemical Engineering Science*, vol. 68, no. 1, pp. 202–214.

LIU, G.-S. / ZHOU, Y. / YANG, H.-D. (2017): Minimizing energy consumption and tardiness penalty for fuzzy flow shop scheduling with state-dependent setup time. In: *Journal of Cleaner Production*, vol. 147, pp. 470–484.

LIU, Y. / DONG, H. / LOHSE, N. / PETROVIC, S. / GINDY, N. (2014a): An investigation into minimising total energy consumption and total weighted tardiness in job shops. In: *Journal of Cleaner Production*, vol. 65, pp. 87–96.

Liu, Y. / Dong, H. / Lohse, N. / Petrovic, S. / Gindy, N. (2014b): An investigation into minimising total energy consumption and total weighted tardiness in job shops. In: *Journal of Cleaner Production*, vol. 65, pp. 87–96.

Liu, Z. / Lee, W.-C. / Wang, J.-Y. (2016): Resource consumption minimization with a constraint of maximum tardiness on parallel machines. In: *Computers & Industrial Engineering*, vol. 97, pp. 191–201.

Lönnberg, M. (2007): Variable Speed Drives for energy savings in hospitals. In: *World Pumps*, vol. o. Jg. No. 494, pp. 20–24.

Lourenço, H. R. / Martin, O. C. / Stützle, T. (2003): Iterated local search. In: *Handbook of metaheuristics*. Springer, pp. 320–353.

Low, C. / Hsu, C.-J. / Su, C.-T. (2008): A two-stage hybrid flowshop scheduling problem with a function constraint and unrelated alternative machines. In: *Computers & Operations Research*, vol. 35, no. 3. Part Special Issue: New Trends in Locational Analysis, pp. 845–853.

Luo, H. / Du, B. / Huang, G. / Chen, H. / Li, X. (2013): Hybrid flow shop scheduling considering machine electricity consumption cost. In: *International Journal of Production Economics*, vol. 146, no. 2, pp. 423–439.

Luo, J. / Fujimura, S. / El Baz, D. / Plazolles, B. (2018): GPU based parallel genetic algorithm for solving an energy efficient dynamic flexible flow shop scheduling problem. In: *Journal of Parallel and Distributed Computing*,

Mansouri, S. A. / Aktas, E. (2016): Minimizing energy consumption and makespan in a two-machine flowshop scheduling problem. In: *Journal of the Operational Research Society*, vol. 67, no. 11, pp. 1382–1394.

Mansouri, S. A. / Aktas, E. / Besikci, U. (2015): Green scheduling of a two-machine flowshop: Trade-off between makespan and energy consumption. In: *European Journal of Operational Research*, vol. 248, no. 3, pp. 772–788.

Marichelvam, M. K. / Prabaharan, T. / Yang, X. S. (2013): A discrete firefly algorithm for the multi-objective hybrid flowshop scheduling problems. In: *IEEE transactions on evolutionary computation*, vol. 18, no. 2, pp. 301–305.

Marler, R. T. / Arora, J. S. (2004): Survey of multi-objective optimization methods for engineering. In: *Structural and multidisciplinary optimization*, vol. 26, no. 6, pp. 369–395.

Mashaei, M. / Lennartson, B. (2013): Energy Reduction in a Pallet-Constrained Flow Shop Through On-Off Control of Idle Machines. In: *IEEE Transactions on Automation Science and Engineering*, vol. 10, pp. 45–56.

Mathaba, T. / Xia, X. / Zhang, J. (2014): Analysing the economic benefit of electricity price forecast in industrial load scheduling. In: *Electric Power Systems Research*, vol. 116, pp. 158–165.

Mavrotas, G. (2009): Effective implementation of the ε-constraint method in multi-objective mathematical programming problems. In: *Applied mathematics and computation*, vol. 213, no. 2, pp. 455–465.

Mecrow, B. / Jack, A. (2008): Efficiency trends in electric machines and drives. In: *Energy Policy*, vol. 36, no. 12. Foresight Sustainable Energy Management and the Built Environment Project, pp. 4336–4341.

Meng, L. / Zhang, C. / Shao, X. / Ren, Y. (2019): MILP models for energy-aware flexible job shop scheduling problem. In: *Journal of Cleaner Production*, vol. 210, pp. 710–723.

Minella, G. / Ruiz, R. / Ciavotta, M. (2008): A review and evaluation of multi-objective algorithms for the flowshop scheduling problem. In: *INFORMS Journal on Computing*, vol. 20, no. 3, pp. 451–471.

Mitra, S. / Grossmann, I. / Pinto, J. / Arora, N. (2012): Optimal production planning under time-sensitive electricity prices for continuous power-intensive processes. In: *Computers and Chemical Engineering*, vol. 38, pp. 171–184.

Modos, I. / Sucha, P. / Hanzalek, Z. (2017): Algorithms for robust production scheduling with energy consumption limits. In: *Computers & Industrial Engineering*, vol. 112, pp. 391–408.

Mokhtari, H. / Abadi, I. N. K. (2013): Scheduling with an outsourcing option on both manufacturer and subcontractors. In: *Computers & Operations Research*, vol. 40, no. 5, pp. 1234–1242.

Mouzon, G. / Yildirim, M. B. (2008): A framework to minimise total energy consumption and total tardiness on a single machine. In: *International Journal of Sustainable Engineering*, vol. 1, no. 2, pp. 105–116.

Naber, A. / Kolisch, R. (2014): MIP models for resource-constrained project scheduling with flexible resource profiles. In: *European Journal of Operational Research*, vol. 239, no. 2, pp. 335–348.

Naderi, B. / Ruiz, R. / Zandieh, M. (2010): Algorithms for a realistic variant of flowshop scheduling. In: *Computers & Operations Research*, vol. 37, no. 2, pp. 236–246.

Nagasawa, K. / Ikeda, Y. / Irohara, T. (2015): Robust flow shop scheduling with random processing times for reduction of peak power consumption. In: *Simulation Modelling Practice and Theory*, vol. 59, pp. 102–113.

Nasiri, M. M. / Abdollahi, M. / Rahbari, A. / Salmanzadeh, N. / Salesi, S. (2018): Minimizing the energy consumption and the total weighted tardiness for the flexible flowshop using NSGA-II and NRGA. In: *Journal of Industrial and Systems Engineering*, vol. 11, no. Special issue: 14th International Industrial Engineering Conference, pp. 150–162.

Nawaz, M. / Enscore, E. E. / Ham, I. (1983): A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem. In: *Omega*, vol. 11, no. 1, pp. 91–95.

Neufeld, J. / Gupta, J. / Buscher, U. (2015): A comprehensive review of flowshop group scheduling literature. In: *Computers & Operations Research*, vol. 70, pp. 56–74.

Nghiem, T. / Behl, M. / Pappas, G. / Mangharam, R. (2011): Green scheduling: Scheduling of control systems for peak power reduction. In: *Green Computing Conference and Workshops*. IGCC, pp. 1–8.

Nilakantan, J. / Huang, G. / Ponnambalam, S. (2015): An investigation on minimizing cycle time and total energy consumption in robotic assembly line systems. In: *Journal of Cleaner Production*, vol. 90, pp. 311–325.

Nilsson, K. / Söderström, M. (1993): Industrial applications of production planning with optimal electricity demand. In: *Applied Energy*, vol. 46, no. 2, pp. 181–192.

Nolde, K. / Morari, M. (2010): Electrical load tracking scheduling of a steel plant. In: *Computers & Chemical Engineering*, vol. 34, no. 11, pp. 1899–1903.

Pan, C.-H. (1997): A study of integer programming formulations for scheduling problems. In: *International Journal of Systems Science*, vol. 28, no. 1, pp. 33–41.

Pan, Q.-K. / Ruiz, R. / Alfaro-Fernandez, P. (2017): Iterated search methods for earliness and tardiness minimization in hybrid flowshops with due windows. In: *Computers & Operations Research*, vol. 80, pp. 50–60.

Pan, Q.-K. / Wang, L. (2008): No-idle permutation flow shop scheduling based on a hybrid discrete particle swarm optimization algorithm. In: *The International Journal of Advanced Manufacturing Technology*, vol. 39, no. 7-8, pp. 796–807.

Pinedo, M. (2012): Scheduling. Volume 5. Springer.

Qi, X. (2009): Two-stage production scheduling with an option of outsourcing from a remote supplier. In: *Journal of Systems Science and Systems Engineering*, vol. 18, no. 1, pp. 1–15.

Qi, X. (2011): Outsourcing and production scheduling for a two-stage flow shop. In: *International Journal of Production Economics*, vol. 129, pp. 43–50.

Quadt, D. / Kuhn, H. (2007): A taxonomy of flexible flow line scheduling procedures. In: *European Journal of Operational Research*, vol. 178, no. 3, pp. 686–698.

Rager, M. / Gahm, C. / Denz, F. (2014): Energy-oriented scheduling based on Evolutionary Algorithms. In: *Computers & Operations Research*, vol. 54, pp. 218–231.

Ramezanian, R. / Sanami, S. F. / Nikabadi, M. S. (2017): A simultaneous planning of production and scheduling operations in flexible flow shops: case study of tile industry. In: *The International Journal of Advanced Manufacturing Technology*, vol. 88, no. 9-12, pp. 2389–2403.

Reyes-Sierra, M. / Coello, C. (2006): Multi-Objective Particle Swarm Optimizers: A Survey of the State-of-the-Art. In: *International Journal of Computational Intelligence Research*, vol. 2, pp. 287–308.

Ribas, I. / Leisten, R. / Framinan, J. M. (2010): Review and classification of hybrid flow shop scheduling problems from a production system and a solutions procedure perspective. In: *Computers & Operations Research*, vol. 37, no. 8, pp. 1439–1454.

Rieseberg, S. / Wörlen, C. (2013): Wachsende Strompreisunterschiede zwischen Gewerbe- und Privatkunden - Marktversagen oder unfaire Abgabenbelastung? Kurzgutachten. Arepo Consult Berlin.

Roll-Hansen, N. (2009): Why the distinction between basic (theoretical) and applied (practical) research is important in the politics of science. London School of Economics and Political Science.

Ruiz, R. / Allahverdi, A. (2009): Minimizing the bicriteria of makespan and maximum tardiness with an upper bound on maximum tardiness. In: *Computers & Operations Research*, vol. 36, no. 4, pp. 1268–1283.

Ruiz, R. / Maroto, C. (2006): A genetic algorithm for hybrid flowshops with sequence dependent setup times and machine eligibility. In: *European Journal of Operational Research*, vol. 169, no. 3, pp. 781–800.

Ruiz, R. / Vazquez-Rodriguez, J. A. (2010): The hybrid flow shop scheduling problem. In: *European Journal of Operational Research*, vol. 205, no. 1, pp. 1–18.

Saidur, R. (2009): Energy consumption, energy savings, and emission analysis in Malaysian office buildings. In: *Energy Policy*, vol. 37, no. 10, pp. 4104–4113.

Saidur, R. / Rahim, N. / Ping, H. / Jahirul, M. / Mekhilef, S. / Masjuki, H. (2009): Energy and emission analysis for industrial motors in Malaysia. In: *Energy Policy*, vol. 37, no. 9, pp. 3650–3658.

Santosa, B. / Riyanto, O. A. W. (2016): Hybrid differential evolution-variable neighborhood search. In: *Lecture Notes in Computer Science*. Volume 9712 LNCS, pp. 587–598.

Saunders, M. / Lewis, P. / Thornhill, A. (2019): Research methods for business students. 8th edition. Harlow, United Kingdom: Pearson Education Limited. Chap. 4.

Sawik, T. (2006): Hierarchical approach to production scheduling in make-to-order assembly. In: *International Journal of Production Research*, vol. 44, no. 4, pp. 801–830.

Sawik, T. (2007): A lexicographic approach to bi-objective scheduling of single-period orders in make-to-order manufacturing. In: *European Journal of Operational Research*, vol. 180, no. 3, pp. 1060–1075.

Scholz, D. (2018): Multikriterielle Optimierung. In: *Optimierung interaktiv*. Springer. Chap. 7, pp. 169–187.

Schulz, S. (2019): A genetic algorithm to solve the hybrid flow shop scheduling problem with subcontracting options and energy cost consideration. In: *Advances in Intelligent Systems and Computing*, vol. 854, pp. 263–273.

Schulz, S. / Buscher, U. / Shen, L. (2020): Multi-objective hybrid flow shop scheduling with variable discrete production speed levels and time-of-use energy prices. In: *Journal of Business Economics*, pp. 1–29.

Schulz, S. (2018): A Multi-criteria MILP Formulation for Energy Aware Hybrid Flow Shop Scheduling. In: *Operations Research Proceedings 2016*. Springer, pp. 543–549.

Schulz, S. / Apelmeier, S. / Buscher, U. (2017): Hybrid Flow Shop Scheduling with Subcontracting Options and Time-Depending Energy Costs. In: *Logistikmanagement - Beiträge zur LM 2017*. Ed. by R. O. Large / N. Kramer / A.-K. Radig / M. Schäfer / A. Sulzbach. Stuttgart, pp. 163–170.

Schulz, S. / Neufeld, J. S. / Buscher, U. (2019): A multi-objective iterated local search algorithm for comprehensive energy-aware hybrid flow shop scheduling. In: *Journal of Cleaner Production*, vol. 224, pp. 421–434.

Shen, L. / Mönch, L. / Buscher, U. (2013): An iterative approach for the serial batching problem with parallel machines and job families. In: *Annals of Operations Research*, vol. 206, no. 1, pp. 425–448.

Smil, V. (2016): Embodied energy: Mobile devices and cars [Numbers Don't Lie]. In: *IEEE Spectrum*, vol. 53, no. 5, pp. 26–26.

Sun, Z. / Li, L. / Fernandez, M. / Wang, J. (2014): Inventory control for peak electricity demand reduction of manufacturing systems considering the tradeoff between production loss and energy savings. In: *Journal of Cleaner Production*, vol. 82, pp. 84–93.

Tan, Y. / Huang, Y. / Liu, S. (2013): Two-Stage Mathematical Programming Approach for Steelmaking Process Scheduling Under Variable Electricity Price. In: *International Journal of Iron and Steel Research*, vol. 20, no. 7, pp. 1–8.

Tang, D. / Dai, M. / Salido, M. A. / Giret, A. (2016): Energy-efficient dynamic scheduling for a flexible flow shop using an improved particle swarm optimization. In: *Computers in Industry*, vol. 81, pp. 82–95.

Tang, L. / Che, P. / Liu, J. (2012): A stochastic production planning problem with nonlinear cost. In: *Computers & Operations Research*, vol. 39, no. 9, pp. 1977–1987.

Triantaphyllou, E. (2000): Multi-Criteria Decision Making Methods. In: *Multi-criteria Decision Making Methods: A Comparative Study*. Boston, MA: Springer US, pp. 5–21. isbn: 978-1-4757-3157-6.

Wang, J. / Qiao, F. / Zhao, F. / Sutherland, J. W. (2016): Batch scheduling for minimal energy consumption and tardiness under uncertainties: A heat treatment application. In: *CIRP Annals*, vol. 65, no. 1, pp. 17–20.

Wang, S. / Wang, X. / Yu, J. / Ma, S. / Liu, M. (2018): Bi-objective identical parallel machine scheduling to minimize total energy consumption and makespan. In: *Journal of Cleaner Production*, vol. 193, pp. 424–440.

Wang, S. / Xiao, X. / Li, F. / Wang, C. (2010): Improved hybrid discrete PSO. In: *2010 8th World Congress on Intelligent Control and Automation*, pp. 4065–4068.

Wauters, T. / Kinable, J. / Smet, P. / Vancroonenburg, W. / Vanden Berghe, G. / Verstichel, J. (2016): The Multi-Mode Resource-Constrained Multi-Project Scheduling Problem. In: *Journal of Scheduling*, vol. 19, no. 3, pp. 271–283.

Wu, X. / Sun, Y. (2018): A green scheduling algorithm for flexible job shop with energy-saving measures. In: *Journal of Cleaner Production*, vol. 172, pp. 3249 – 3264.

Xu, F. / Weng, W. / Fujimura, S. (2014): Energy-Efficient Scheduling for Flexible Flow Shops by Using MIP. In: *IIE Annual Conference and Expo 2014*. Institute of Industrial Engineers., pp. 1040 – 1048.

Yan, J. / Li, L. / Zhao, F. / Zhang, F. / Zhao, Q. (2016): A multi-level optimization approach for energy-efficient flexible flow shop scheduling. In: *Journal of Cleaner Production*, vol. 137, pp. 1543 – 1552.

Yang, W. / Wang, J. / Niu, T. / Du, P. (2020): A novel system for multi-step electricity price forecasting for electricity market management. In: *Applied Soft Computing*, vol. 88, p. 106029.

Yenisey, M. M. / Yagmahan, B. (2014): Multi-objective permutation flow shop scheduling problem: Literature review, classification and current trends. In: *Omega*, vol. 45, pp. 119 – 135.

Yu, C. / Semeraro, Q. / Matta, A. (2018): A genetic algorithm for the hybrid flow shop scheduling with unrelated machines and machine eligibility. In: *Computers & Operations Research*, vol. 100, pp. 211 – 229.

Yusta, J. / Torres, F. / Khodr, H. (2010): Optimal methodology for a machining process scheduling in spot electricity markets. In: *Energy Conversion and Management*, vol. 51, no. 12, pp. 2647 – 2654.

Zanoni, S. / Bettoni, L. / Glock, C. (2014): Energy implications in a two-stage production system with controllable production rates. In: *International Journal of Production Economics*, vol. 149, pp. 164 – 171.

Zavadskas, E. K. / Turskis, Z. / Kildiene, S. (2014): State of art surveys of overviews on MCDM/MADM methods. In: *Technological and economic development of economy*, vol. 20, no. 1, pp. 165 – 179.

Zeng, Z. / Hong, M. / Man, Y. / Li, J. / Zhang, Y. / Liu, H. (2018): Multi-object optimization of flexible flow shop scheduling with batch process — Consideration total electricity consumption and material wastage. In: *Journal of Cleaner Production*, vol. 183, pp. 925–939.

Zhang, H. / Zhao, F. / Fang, K. / Sutherland, J. (2014): Energy-conscious flow shop scheduling under time-of-use electricity tariffs. In: *CIRP Annals - Manufacturing Technology*, vol. 63, no. 1, pp. 37–40.

Zhang, Q. / Li, H. (2007): MOEA/D: A multiobjective evolutionary algorithm based on decomposition. In: *IEEE Transactions on evolutionary computation*, vol. 11, no. 6, pp. 712–731.

Zhang, R. / Chiong, R. (2016): Solving the energy-efficient job shop scheduling problem: a multi-objective genetic algorithm with enhanced local search for minimizing the total weighted tardiness and total energy consumption. In: *Journal of Cleaner Production*, vol. 112, pp. 3361–3375.

Zheng, H. / Wang, L. (2015): Reduction of carbon emissions and project makespan by a Pareto-based estimation of distribution algorithm. In: *International Journal of Production Economics*, vol. 164, pp. 421–432.

Zheng, X.-l. / Wang, L. (2016): A two-stage adaptive fruit fly optimization algorithm for unrelated parallel machine scheduling problem with additional resource constraints. In: *Expert Systems with Applications*, vol. 65, pp. 28–39.

Zhou, B. / Liu, W. (2019): Energy-efficient multi-objective scheduling algorithm for hybrid flow shop with fuzzy processing time. In: *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, vol. 233, no. 10, pp. 1282–1297.

Ziesing, H.-J. (2018): Energieverbrauch in Deutschland im Jahr 2017. Technical report. Arbeitsgemeinschaft Energiebilanzen e. V.

Zitzler, E. / Thiele, L. (1998): Multiobjective Optimization Using Evolutionary Algorithms - A Comparative Study. In: *Parallel Problem Solving from Nature V*. Amsterdam: Springer, pp. 292–301.